AD-A187 703 THE IMPLEMENTATION OF A PLANT AND MINOR PROPERTY ACCOUNTING SYSTEM FOR THE NAVAL POSTGRADUATE SCHOOL(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W T KEY ET AL. SEP 87 1/3

UNCLASSIFIED F/G 5/1 NL

1.0
1.1
1.25
1.4
1.6

[illegible]

②

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE IMPLEMENTATION OF A PLANT AND MINOR PROPERTY ACCOUNTING SYSTEM FOR THE NAVAL POSTGRADUATE SCHOOL

by

William Tigner Hay
and
Bruce Alan Whitehouse, II

September 1987

Thesis Advisor Daniel R. Dolk

8a NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c ADDRESS (City, State and ZIP Code)	10 SOURCE OF FUNDING NUMBERS			
	PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
THE IMPLEMENTATION OF A PLANT AND MINOR PROPERTY ACCOUNTING SYSTEM FOR THE NAVAL POSTGRADUATE SCHOOL (u)

12 PERSONAL AUTHOR(S)
Key, William Tiener and Whitehouse, Bruce Alan II

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM TO	1987 September	235

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Automated Inventory, Plant and Minor Property, Inventory Database System, Automated Data Dictionary

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

In October 1986, the Naval Postgraduate School was directed to centralize the accounting and control of minor property (approximately 60,000 items) and to locate that function in the Supply Department. The magnitude of this undertaking suggested that some sort of automated system be employed to assist in the task.

The objective of this study was to implement a prototype automated system to support the control and accounting of both plant and minor property at the Naval postgraduate School. The effort was based on a Requirements Definition and System Specification set forth in a thesis by Ross and Smith in March 1987. The study includes a description of the implementation

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
[✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT ☐ DTIC USERS	Unclassified

22a NAME OF RESPONSIBLE INDIVIDUAL	22b TELEPHONE (Include Area Code)	22c OFFICE SYMBOL
Prof. Daniel R. Dolk	(408) 646-2260	Code 54DK

19. ABSTRACT(CONTINUED)
effort, prototype code written in dBase III Plus, and a User's Quick Reference. The basic structure of the code and the data base design should be applicable throughout the U.S. Navy Supply system.

The Implementation of a Plant and Minor Property Accounting System for the Naval Postgraduate School

by
William Tigner Key
Commander, United States Navy
B.A., Georgia State University, 1972

and

Bruce Alan Whitehouse, II
Captain, United States Marine Corps
B.S., Lyndon State College, 1979

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the
NAVAL POSTGRADUATE SCHOOL
September 1987

Authors: [illegible]
William Tigner Key

[illegible]
Bruce Alan Whitehouse, II

Approved by: [illegible]
Daniel R. Dolk, Thesis Advisor

[illegible]
Judith H. Lind, Second Reader

[illegible]
Willis R. Greer, Jr., Chairman,
Department of Administrative Sciences

[illegible]
James M. Fremgen
Acting Dean of Information and Policy Sciences

ABSTRACT

In October 1986, the Naval Postgraduate School was directed to centralize the accounting and control of minor property (approximately 60,000 items) and to locate that function in the Supply Department. The magnitude of this undertaking suggested that some sort of automated system be employed to assist in the task.

The objective of this study was to implement a prototype automated system to support the control and accounting of both plant and minor property at the Naval Postgraduate School. The effort was based on a Requirements Definition and System Specification set forth in a thesis by Ross and Smith in March 1987. The study includes a description of the implementation effort, prototype code written in dBase III Plus, and a User's Quick Reference. The basic structure of the code and data base design should be applicable throughout the U.S. Navy Supply system. [illegible]

[illegible] data ba[illegible] Systems

TABLE OF CONTENTS

LIST OF ACRONYMS

ADPE	Automatic Data Processing Equipment
CPU	Central Processing Unit
DBA	Data Base Administrator
DBMS	Data Base Management System
DARIC	Defense Automation Resources Management System
DIPEC	Defense Industrial Plant Equipment Center
DD	Defense Document
DOD	Department of Defense
DRMO	Defense Reutilization Management Office
IE	Industrial Equipment
IG	Inspector General
IPE	Industrial Plant Equipment
IPO	Input/Output/Process
KB	Kilobyte
MB	Megabyte
MCD	Material Control Division
NAVCOMPT	Navy Comptroller
NID	Navy Identification Number
NPS	Naval Postgraduate School
PMB	Property Management Branch
PMPS	Plant and Minor Property System
RAM	Random Access Memory
SF	Standard Form
SOP	Standard Operating Procedure
WS	Work Sheet

LIST OF FIGURES

I. INTRODUCTION

A. BACKGROUND

Virtually every U.S. Navy shore command owns some physical assets that are not expended when used--desks, typewriters, file cabinets, tool kits, calculators, etc. Some assets are relatively inexpensive (for example, a desk lamp), while others have a high dollar value (for example, a mini-computer or specialized test equipment). In Navy parlance these assets are known as "minor property" and "plant property" respectively. The distinction between the two is marked by dollar value. Generally speaking minor property costs less than $5,000 and plant property more than $5,000 [NAVCOMPT MANUAL, VOL 3, Chapter 6]. Navy regulations and good management practice dictate that these assets be controlled and accounted for from the time that they are ordered by the command until they are disposed of or replaced.

At the Naval Postgraduate School (NPS), plant property (about 4,500 items) is controlled centrally by the school's Supply Department, while minor property (as many as 60,000 items) is controlled at the academic department level. In both cases, control is performed manually using a variety of Navy and U. S. government forms and procedures. [Ross and Smith, 1987, pp. 16-25]

In October 1986, NPS was directed to centralize the accounting of minor property and to locate that function in the Supply Department. The magnitude of this undertaking, as well as common sense, suggested that some sort of automation be employed to assist in the task. Two students in the Information Systems curriculum (Ross and Smith) took on the problem as a thesis effort. The goals of their project were to assist the command in complying with the centralization directive, and to help improve property administration in general. Ross and Smith concluded that an automated data base management system would accomplish both objectives [Ross and Smith, 1987, Chapter 3]. The Ross-Smith thesis provided a detailed system specification, initial data dictionary, recommended file structures, and hardware specification for a database management system that would fully automate both plant and minor property administration.

The intent of the effort discussed in this thesis is to <u>implement</u> the recommendations of the Ross-Smith study in a prototype system. Such a prototype could be used to verify Ross's and Smith's conclusions and refine user requirements further. A prototype methodology was elected due to the sheer magnitude of the task, the relatively short time available, and user unfamiliarity with computer technology. Initial prototype tools included sample screens and sample functions (reports, sorts, etc.), followed by actual coding of selected modules of the system.

It should be noted at the outset that familiarity with Ross's and Smith's research is essential for any follow-on research or programming effort. Such familiarity also will be helpful to the casual reader of this document, but is not critical.

B. OBJECTIVES

The objectives of our effort are:

1. To assist NPS in meeting the requirements of the "centralization" directive.
2. To implement a practical prototype system to support the NPS Supply Department in the administration of plant and minor property.
3. To verify that the Ross-Smith specification is both complete and accurate; i.e., that it covers all user requirements and that requirements are specified correctly.
4. To investigate the use of the application package dBase III Plus with a very complex and potentially very large data base.

C. APPLICABILITY

The basic tasks of the Key-Whitehouse effort were to verify the findings of Ross and Smith, and to implement a prototype of an automated system which assists in accounting and controlling plant and minor property at the Naval Postgraduate School.

The recommendations made by Ross and Smith, combined with the results of this present effort, should be applicable throughout the U. S. Navy, as regulations for handling plant and minor Property are largely standardized.

D. RESEARCH QUESTIONS

The following questions have been addressed in this study. Conclusions resulting from this study are included in Chapter V.

1. How should an automated property accounting system be designed in order to avoid data redundancy and to maximize performance?
2. What design characteristics can be included to minimize insertion and deletion anomalies?
3. What error checks are needed in a system that will be operated by personnel at the clerk/typist level, and how can they be implemented when programming in the dBase III Plus data base management system?
4. Can an on-line data dictionary be designed that will support both the user and a follow-on programmer?
5. Is dBaseIII Plus adequate for a data base with 65,000 records and that is expected to grow as large as 130,000?
6. Is such an accounting system applicable for Navy-wide use?

E. METHODOLOGY

A traditional system software development cycle typically consists of some variation of these generic phases: requirements definition, design, coding, testing, delivery, and maintenance. In this traditional cycle there is often a significant time delay between the initial requirements analysis and the delivery of the operational system. Unfortunately, after requirements definition is finished, it is not unusual for a designer or programmer to consult the user only when some specification is unclear. Indeed it is not uncommon for the user to see the system for the first time at final delivery.

This approach has deservedly come under frequent attack in recent literature [e.g., Harrison, 1985, pp. 22-25; Boarly, 1983, pp. 36-42]. A popular solution to this dilemma is software prototyping, in one of its various forms. Given the characteristics of the development environment prototyping was selected as the basic methodology for this study for several reasons.

First, the system requirements analysts (Ross and Smith) and the programming team (Hoy and Whitehouse) were unable to work together on this project and Ross and Smith were not available for consultation or clarification of their requirements document. Thus it was necessary for NPS Supply Department personnel to explain the contents of the Ross-Smith document, portions of which they had never seen, and which was written in unfamiliar terms (e.g., data elements data flow diagrams, IPO charts, inter alia).

Second, the project required performing a coding task while working against an inflexible deadline (the academic calendar). It is axiomatic that software projects take longer and cost more than originally estimated. Thus it was important to choose a methodology that would allow great flexibility and clear communication to a follow-on effort should the code not be 100 percent complete when time ran out.

Third, the original estimates of the data base size were as high as 65,000 items, and the system was to be implemented on a microcomputer. Thus it was important to choose an approach that could act as an ongoing feasibility study and allow the developers to identify technical blocks in the critical path at the earliest opportunity .

Finally, the acquisition and disposition process at NPS is extremely complex and replete with opportunities for significant misunderstanding by the programmers.

A prototyping methodology could be used to verify the Ross-Smith document and also to ensure the authors' understanding of the problem as coding progressed. If necessary, the project could be left in a form that could be easily continued by others. A significant benefit of this approach is that not only were the authors able to verify and clarify what was contained in the Ross-Smith effort, but also areas that had been overlooked or changed were discovered. These areas almost certainly would have remained uncovered until after final delivery had the traditional approach been used.

Specifics regarding prototyping approach and changes to the original requirements documentation are included in Chapter III.

F. STRUCTURE OF THE THESIS

The remainder of the thesis is organized chronologically. It describes where our effort began, based on the Ross-Smith work, followed by what was done for this study, where the development was stopped, and what remains to be done.

Chapter II describes the environment for which research was performed. Navy and NPS requirements are discussed regarding centralization of minor property administration and automated systems. The application environment, initial assumptions and constraints, and characteristics of the organizational environment are then described. Finally, a brief narrative description of the equipment acquisition and disposition process is provided.

Chapter III provides prototyping strategy, details of data dictionary and data base design, and a discussion of data integrity issues and the role of the data base administrator, particularly as they apply to the Material Control Division (MCD) of the Supply Department.

Chapter IV covers the programming implementation and includes heirarchy charts, user interfaces, and sample queries with responses.

Chapter V deals with what remains to be accomplished before the system can be properly called operational. It covers both general and specific tasks for the user and follow-on programmers. Chapter V concludes with comments regarding the original research questions.

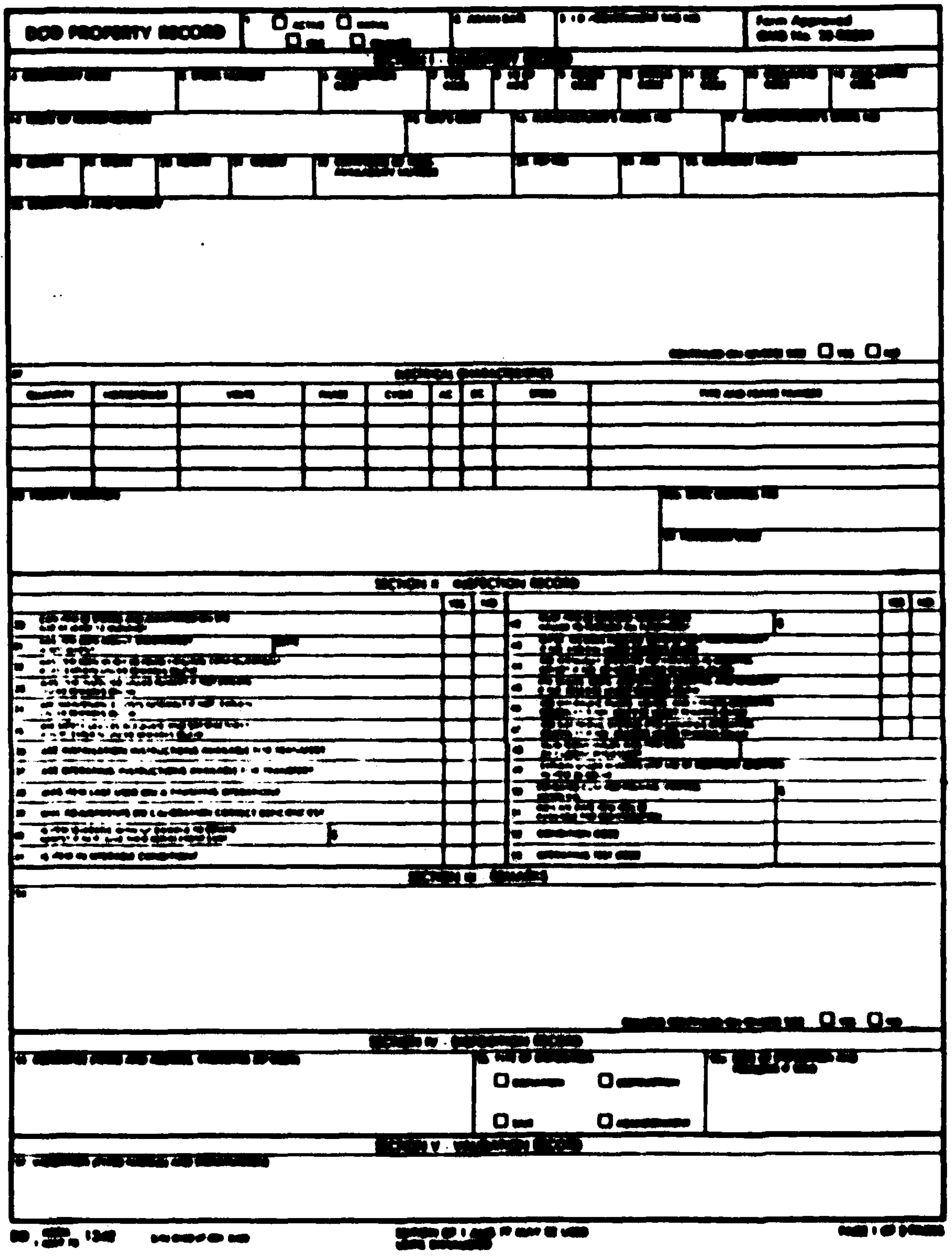
II. THE NPS PROPERTY ACCOUNTING ENVIRONMENT

A. THE PROPERTY ACQUISITION PROCESS AT NPS

The Acquisition process at NPS consists of three phases. These occur chronologically, prior to a piece of equipment being considered "in hand and on file". The goal of the process is to complete a form DD 1342 Property Record (Figure 2.1). The following summary of the acquisition phases include the corresponding government form used in that phase, noted in parentheses (see Ross-Smith Appendix B for detailed description of forms).

The first phase begins when MCD is notified that a piece of equipment has been ordered. Only a few facts are known at this point: the Naval Identification (NID) number (when assigned), manufacturer, and model number (form WS 1342). The second phase begins upon arrival of the equipment. A few additional pieces of data now become available: e.g., consignor, date received, requisition number (form WS 1342(+)). For the third phase, the piece of equipment is sent to the destination department. There the equipment is unpacked, and the final missing pieces of data can be provided to MCD: manufacturer's serial number, location, power code, size, etc. (form DD 1342).

Once completed, MCD's files of the DD 1342s comprise the present database that is queried for the types of questions



DOD PROPERTY RECORD

[illegible]

Figure 2.1 DD 1342 DOD Property Record

noted in section D. below. This process is currently carried out for plant property only and is essentially a manual process performed by a single individual in MCD.

B. THE PROPERTY DISPOSITION PROCESS AT NPS

The disposition process for plant Property is somewhat more complicated than the acquisition process. There are five types of dispositions:

- Industrial Plant Equipment (IPE)
- Automatic Data Processing Equipment (ADPE)
- Transfers
- Trade-ins
- Local dispositions

Each type is handled in a slightly different manner. Differences include (1) what organizations are notified that a particular piece of equipment is available for reutilization, and (2) the amount of time that a piece of equipment is retained before local disposition is authorized.

Both the IPE and the ADPE items identified as excess items may be held as long as 201 days, according to the directions issued by the Defense Industrial Plant Equipment Center (DIPEC) and the Defense Automated Resources Information Center (DARIC). During that time either of these organizations may issue transfer instructions. Should this be done, the holding organization, NPS, transfers the equipment to the organization designated in the instructions.

When the plant Property Administration Office at NPS receives verification that the organization to which the item has been shipped has received the shipment, the equipment is removed from the NPS accounting records.

Trade-in dispositions occur when a department actually trades in an old piece of equipment for a new one. The proper organizations again must be notified and the item is removed from the records. Local dispositions include, for the most part, items that do not fit into one of the other four catagories. After the appropriate time has elapsed, the item is taken to Fort Ord's Defense Reutilization Management Office for processing. When the equipment has been accepted there it may be expunged from NPS's records.

Minor property does not have to be reported to any of the above organizations and thus does not as yet have a formal disposition process. However, it is expected that minor property will be made available for local reutilization for a 21-day period. At the end of that time it will be disposed of locally.

C. FUNCTIONAL REQUIREMENTS

The purpose of the Ross-Smith study was to produce design specifications for an automated system that would support the centralized management and control of plant and minor property at NPS. Such a system must meet the functional needs of MCD while also satisfying requirements of higher authority as stated in NAVCOMPT Manual Vol 3 Chapter 6. In general terms the new system must:

1. Create auditable records on all acquisitions, disposals, and transfers.
2. Support preparation of form DD 1342, DOD Property Record.
3. Support the preparation of a plant or minor property report in accordance with NAVCOMPT 274.
4. Provide inventory listings by department, item, inventory date, or other selected sort criteria.
5. Compare inventory listings from departments to determine lost, missing, or stolen property.
6. Prevent unauthorized access to the database [Ross, 1987, pp. 74-75].

Functional requirements as stated by Ross and Smith are accurate, complete, and still valid as of this writing.

D. CHARACTERISTICS OF THE APPLICATION ENVIRONMENT

The application environment can be characterized by describing typical queries, data base volume, and update and retrieval activity; a brief summary of each follows.

1. Kinds of Queries to be answered

A wide variety of data base queries must be supported by the proposed automated system. These include such questions as:

-What plant/minor property does the Administrative Science Department hold?

-Where is (some particular) piece of plant/minor property located?

-What dollar value in minor property is held by the Electrical Engineering Department?

-Where is all of the XYZ Corp. equipment (e.g., for maintenance)?

-What dollar value in plant/minor property was lost when Root Hall burned to the ground?

-How big is a certain piece of plant property (e.g., for shipment)?

2. Expected Data Base Volume

The current inventory of NPS plant property includes approximately 4500 items. The minor property inventory may be as high as 60,000 items if all potential items are counted. In view of the existing inventory size and the limited manpower available, once an automated data base is available, the NPS plan is to enter new items into the database as they arrive. As time permits, existing items will be "back-entered".

Data base volume will be affected significantly by management decree determining what items will be included. The cost of accounting and control for every hammer, screwdriver, and pair of scissors would outweigh the benefit by an order of magnitude. The exact items to be included in the data base have not been determined. Where MCD draws its lines will largely determine the volume of data to be handled and the number of records in the data base.

3. Relative Update and Retrieval Activity

The current property accounting system is essentially manual and the NPS Supply Department has historically dealt only with plant property (minor property being accounted for at the department level). Thus an estimate of the update and retrieval activity for the data base would require a department-by-department survey. Even if such an figure were presently available, it would likely be inaccurate with respect to the future. The power of an automated data base and the ease of automated retrieval will almost certainly generate a higher level of data base access activity than is currently experienced. In addition, the factors mentioned in the Expected Data Base Volume section will also affect the update and retrieval frequency.

If however, operations continue to be conducted as they are at present, the ratio of update transactions to query transactions will be about 70/30, respectively. Updates would probably be performed twice daily in "batches", with queries distributed randomly throughout the day.

E. CHARACTERISTICS OF THE ORGANIZATIONAL ENVIRONMENT

The organizational situation in the NPS Supply Department MCD has a major impact on the implementation of the proposed system. During the period of this implementation effort, MCD has undergone significant changes. Originally staffed with five employees to handle acquisitions and dispositions, the staff has shrunk to as few as three persons, including a new Division head. Supervisory personnel have been lost and not replaced, new hires have arrived only to depart after a week, and in general, ths situation has been one of disorder. Complicating matters was the U. S. Navy directive to centralize the accounting and control of minor property. This represented a substantial new task of unknown magnitude and complexity, for which there were no standard operating procedures or corporate memory and experience. Thus we arrived in the midst of organizational turbulence and a significant but ill-defined mission change with minimal structure and assets in place to support it.

F. CONSTRAINTS

Constraints arise when a research project or study is a follow-on to a previous effort. This is especially true when the follow-on project is a continuation of a "real world" development effort (as opposed to a purely academic exercise). For example, Ross and Smith provide a description of the property accounting environment which was verified by the Supply Department users during their research. This description was accepted as being essentially correct and the present study was conducted with that description as a constraint. Additionally, Navy and users' terminology (as well as Ross's and Smith's terminology) was adopted for continuity during the present study rather than creating an additional set of terms.

III. DATABASE DESIGN

A. INTRODUCTION

This chapter addresses specifics of the prototyping approach for this project, describes the data dictionary and the automated system data base and includes discussion of data integrity issues. It concludes with data integrity and data base administration issues relevant to MCD operations.

B. PROTOTYPING GOALS AND STRATEGIES

Chapter I discussed why a prototyping methodology was selected. The following describes the prototyping goals and strategies for this effort.

Although the prototype system was viewed as a vehicle for verifying user requirements and our understanding of the system, it was also intended to serve as a framework for the ultimate, complete system. This required that the prototype be developed in the context of an overall system plan. Consequently a top-down strategy was used to focus the prototyping effort on the major functions of the system: system attributes that would be developed for the user and would demonstrate both the behavior and the feasibility of the entire system. This strategy resulted in the coding of both the acquisition and disposition software modules at the outset (as opposed to doing just one of them in greater detail).

The next important consideration was the level of detail that should be coded into the prototype. The authors' decision was to include the minimum detail necessary to demonstrate the system functions. Efficient use of designer time and rapid feedback were considered more important than robust operation, efficient use of machine resources, or significant error proofing. [Berzins and Berzins, 1986]

As a result of the prototyping goals and environment, the authors reinforced their understanding of the Ross-Smith document, verified the correctness of it, and implemented significant portions of it. New requirement definitions were written for areas overlooked or changed and these were coded where possible. Different prototyping tools were used during various stages and included sample screens, sample menus, sample reports and occasional briefings for the user covering the authors' understanding of a concept or process. Interaction with the user was routine and regular and included interaction with both management (head of MCD) and with the expected primary day-to-day users of the automated system.

C. THE DATA DICTIONARY

Our attempt at an automated data base design began with the data dictionary. Not addressed by the Ross-Smith study, this effort was essentially conducted "from scratch". Consequently, a certain amount of trial and error was involved. From a design standpoint, our challenge was to maintain a very fragile balance between the desire to minimize data redundancy (for storage efficiency and retrieval speed) and a requirement for the maximum amount of cross referencing possible.

The next three subsections describe the data dictionary in its current form. The first covers data dictionary definition, contents, and utility; the second, structure and operation. The final subsection contains a discussion of the motivations and justification for an on-line data dictionary and for our approach.

1. Definition, Contents and Utility

A data dictionary is a software tool that is used to control and manage data elements in a uniform manner [Ross, 1981, pp. 15-38]. It can serve data base administrators, system analysts, software designers, programmers, and users by providing a central repository for information about data resources across organization and application boundaries.

The data dictionary designed as a part of this study is a repository for the source data definitions of the MCD of the Supply Department at NPS, plus text to aid both MCD users and those who might continue this effort. Items defined include non-automated as well as automated data and apply to conventional files as well as to the automated system's data base. The data dictionary contains definitions and cross references to files, programs, formats, and forms.

Normally a data base involves multiple users using the same data [Perry, 1980, p. 43]. Presently, the only user of this data base is the MCD. However, there is no reason why some later version of this system could not be used by other associated organizations, such as the various curricula officers, or organizations other than NPS to keep track of plant and minor property under their control.

2. Structure and Operation of the Data Dictionary

Appendix A contains the code for the on-line data dictionary. The data dictionary system is comprised of five files, or relations.

1. DD_DELEM.DBF: this contains all the data elements, including aliases, full names, a text definition, and the programs, formats, and forms where they appear.
2. DD_FILES.DBF: this file contains an entry for each relation, and a brief description of each, its content, and where each is used.
3. DD_PRGRM.DBF: this lists all program names (both what they are referred to within the coded system and full names as in the system specifications) and the purpose of each program.
4. CALL_PGM.DBF: this lists each program, by nickname, and all other programs it calls, or is called by.

5. DD_CONTA.DBF: this file provides a cross reference of program names and file names.

Figure 3.1 shows the specific fields contained in each of the subdictionaries.

DD_PROGM = PRGNAME, FULLNAME, PURPOSE;

DD_DELEM = NICKNAME, FULLNAME, DEFINITION, FORMAT,
USED_IN;

DD_FILES = FILE_NAME, DESCRIPTN, CONTENTS, USED_BY;

DD_CONTA = PRGNAME, USES;

CALL_PGM = PRGNAME, CALLS, CALLEDBY;

Figure 3.1 Contents of the Data Dictionary Relations (key underlined)

Each of these dictionary files can be updated by using either an on-line menu-driven system called DATADICT.PRG (see Figure 3.2) or standard dBase III Plus commands. Special reports or queries (other than the individual look-ups provided by DATADICT.PRG) must be requested using the dBase III Plus command language. Figure 3.3 shows a such a query and the response. In this example, a programmer has changed the software code in a module (ACQUISIT.PRG) and wants to know what other programs may be affected, i.e., those programs that it calls.

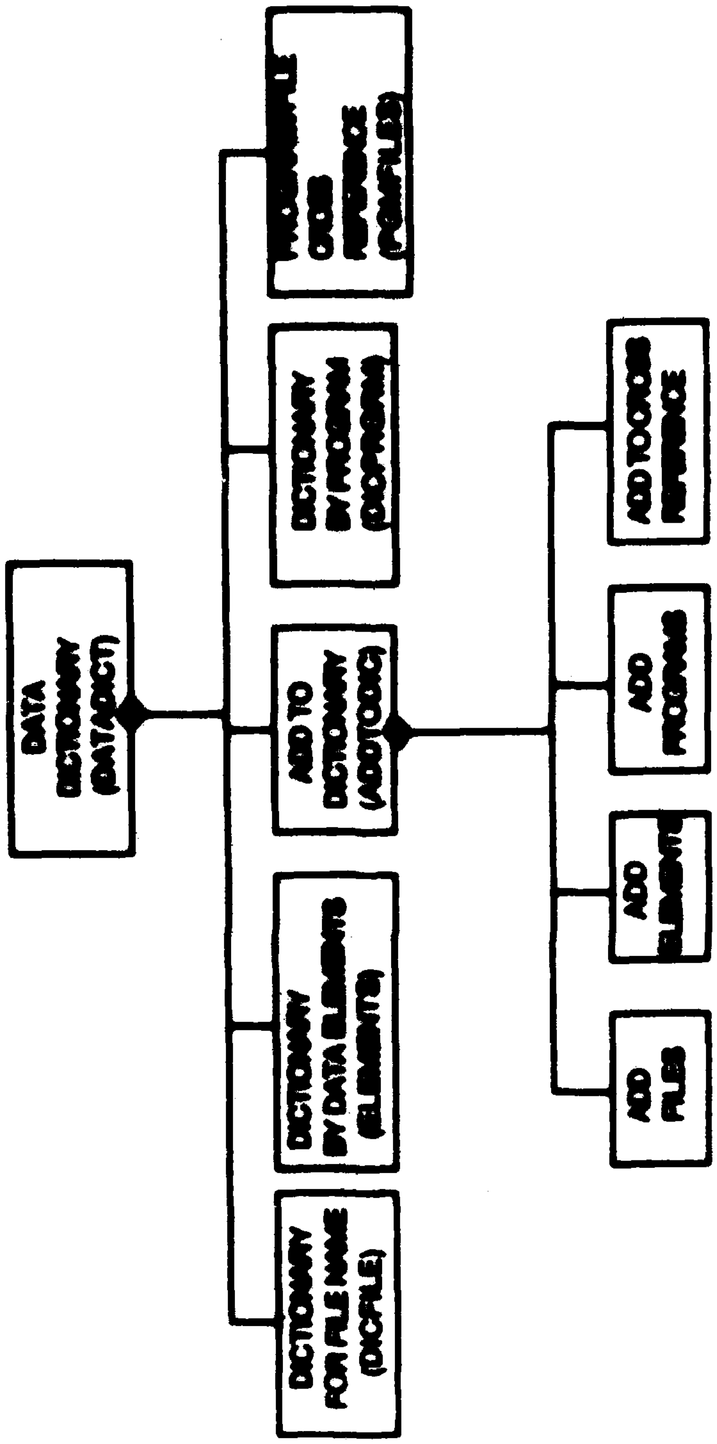

DATA DICTIONARY (DATADICT)
DICTIONARY FOR FILE NAME (DICFILE)
DICTIONARY BY DATA ELEMENTS (ELEMENTS)
ADD TO DICTIONARY (ADDTODIC)
DICTIONARY BY PROGRAM (DICPROGRAM)
PROGRAM/FILE CROSS REFERENCE (PGMFILES)
ADD FILES
ADD ELEMENTS
ADD PROGRAMS
ADD TO CROSS REFERENCE

Figure 3.2 Structure of DATADICT.PRG

```
.USE CALL_PGM

.DISPLAY PGMNAME, CALLS FOR PGMNAME="ACQUISIT"

Record#  PGMNAME        CALLS

      5  ACQUISIT       RQ_RECPT

      6  ACQUISIT       NT_PEN_R

      7  ACQUISIT       MAKE1342

      8  ACQUISIT       PRNT1342

      9  ACQUISIT       VIEW1342

     10  ACQUISIT       INV_MENU

     11  ACQUISIT       PENDMENU
```

Figure 3.3 Data Dictionary Query Using DBase III Commands

3. Design Motivation and Structure Justification

The data dictionary was designed and coded with certain goals for data resource development in mind:

- Providing users with better access to information about the data and the system resources.
- Enhancing the ability of those who develop, maintain, and/or improve the information system.
- Supporting more effective application system design.

Technology has progressed to where the focus is now on making the services of a data dictionary system available on an on-line, interactive basis. This is consistent with industry which has moved toward on-line systems. The following observations underscore the importance of this and why the data dictionary for this effort is built to function on-line.

- The increasing significance of the data dictionary to data administration and ongoing maintenance puts new demands on the data dictionary system and its operation for timely access to information. At the same time, the types of questions asked of the data dictionary are becoming either more specific or more ad hoc in nature. Both of these facts point forcibly in the direction of on-line access.
- The data dictionary is often viewed as a point-in-time reference rather than a static book-of-record catalog. This too makes timely access to current information sources a growing importance [Ross, 1981, pp. 15-38].

No data dictionary system, especially one with extensibility, can predict the exact form of all eventual user reporting requirements. Nor is it possible to know in advance all occurrences of a data element and its relation to other entities in a system. For this reason, a generalized capacity for cross reference mapping is important. The data dictionary as currently designed contains data bases of programs, data elements, data base files (relations) and formats; each contains cross references to the others. For example, each data element entry in the data dictionary contains a listing of every occurrence of that data element in a relation, program, format, or form.

There are many other potential forms such a mapping might take. In general, its options should include the ability to identify one or more occurrences of the first entity type and trace its relationships to all associated occurrences of a second entity type, with occurrence selection optionally restricted by specified criteria such as attribute value or status. If the path between these two types crosses one or more intervening entity types, the option to list occurrences of these intervening types should also be offered. If more than one path exists, the ambiguity must be resolved by the report requester. Presently, the data dictionary system does not have these features. The user must use dBase III Plus to query the data bases directly to achieve this level of mapping.

At present, the data dictionary is functional but sub-optimal from a design and data integrity standpoint. For example, the "FORMAT" field in the DD_DELEM relation could be better designed for both program efficiency and data integrity. This field presently contains up to 20 characters of text that describe the length and data type of a data element. The field should be split into three fields: one containing length (maximum of 2 characters), another for designation of field type--alpha or numeric (1 character), and the third for comments where necessary. Additionally, the structure of the relations could be further normalized, possibly to the point of only two fields per relation. Finally, in several places where the amount of data to be placed in a field varies greatly, the dBase III feature of a "memo" type field is used. There may be a more efficient way of dealing with these variations in field length.

It should also be noted that, in its present form, the data dictionary is of primary use to the programmer. Other entities or relations could be added that would be of value to a data base administrator or other users. For example, fields such as "FILE_POC" or "REF_DOC" could be added to relation "DD_FILES" to store the name or code of the responsible individual and the germane reference document applicable to the maintenance of certain files. An entirely new relation could be created which would contain, for each department, a separate point of contact for equipment classified as ADPE, audio-visual, or labor saving device.

D. THE DATA BASE

The prototyping process resulted in some changes to the Ross-Smith data base design. The most significant of these are discussed below; the remainder are incorporated into the code in Appendices B and C.

Both the acquisition and disposition process logical models required some modification and redefinition which affected the data base. For example, duplication of records in the Pending Dispositions File that do not have a NID number was alleviated by reducing the date termination field (DATE_TERM) to a four character Julian date field and identifying the record by placing the correct date and stock number in the NID number field. Also, several data elements were added to the contents of form DD 1342 (discussed in detail in Chapter IV).

Probably the most significant design changes occurred during the normalization process. Ross-Smith's four original data base files were restructured (normalized) to reduce memory requirements and reduce insertion and deletion anomalies. The process began with an element-by-element analysis of each of the four logical data base files. A separate relation was created to correspond to the occurrences of each of data elements in the logical files. For example, all of the data elements which occurred only in the logical Plant file were placed in one relation (PLANT). The data elements which were found in all four of the logical files were placed together in another relation (PROPERTY). Data elements which were found only in the logical files Pending and D-Pending were placed in a relation called PENDING. Figure 3.4 summarizes this process. Notice that the data elements which were located in the Plant, Minor and D-Pending logical files were further subdivided into two relations (MFG_INFO and ITEM_DES) when it was discovered that ITEM_DES could be indexed on STOCK_NO in addition to NID_NO.

DATA ELEMENT IN LOGICAL FILE(S)	PLACED IN RELATION
Plant (only)	PLANT
Plant and Minor	INV_INFO
All four	PROPERTY
Plant, Minor, and Pending	PURCHASE
Pending and D-Pending	PENDING
D-Pending (only)	PENDING-D
Plant, Minor, and D-Pending	ITEM_DES and MFG_INFO

Figure 3.4 Normalization of the Logical Files

Figure 3.5 shows a breakdown of the original four files into the relations that they consist of. Each relation is now a separate data base ("xxx.dbf" in dBase III). The content (data elements) of each relation is also shown in Figure 3.6.

E. DATA INTEGRITY AND DATA BASE ADMINISTRATION

Control in a data base environment refers to the methods that an organization uses to safeguard its assets, ensure accuracy and reliability of data in its data bases, and enforce adherence to management policies [Perry, 1981, p. 1]. Two of the most important steps toward establishing control in a data base environment are: (1) defining and enforcing data integrity constraints and (2) performing the functions

PLANT FILE	MINOR FILE	PENDING FILE	D-PENDING FILE
PLANT	PROPERTY	PENDING	PEND-D
PROPERTY	PURCHASE	PROPERTY	PROPERTY
PURCHASE	ITEM_DES	PURCHASE	ITEM_DES
ITEM_DES	MFG_INFO		MFG_INFO
MFG_INFO	INV_INFO		
INV_INFO			

Figure 3.5 Logical Files and Relations

PROPERTY	=	NID_NO, MFG_NAME, MFG_SER_NO, NOUN_NAME, MFG_MOD_NO, PROP_CODE, DEPT_CODE
PLANT	=	NID_NO, COMOD_CODE, LENGTH, MFG_CODE, NON_AVL_NO, PWR_CODE, WEIGHT, WIDTH, HEIGHT
PENDING	=	NID_NO, PEND_STAT
PEND-D	=	NID_NO, COND_C_E, DATE_ENT_D, DATE_TERM, PHASE, POC_N, POC_P_N, QTY_E, RPT_NO
MFG_INFO	=	NID_NO, MFG_YEAR, STOCK_NO, TYPE_CODE, COST
ITEM_DES	=	STOCK_NO, PHYS_DESC
PURCHASE	=	NID_NO, CONSIGNOR, DATE_RECV, PO_NO, REQN_NO
INV_INFO	=	NID_NO, INV_DATE, JULIAN, LOCATE

Figure 3.6 Contents of the Data Base Relations

of a data base administrator. The next two subsections discuss each of these in turn.

1. Definition and Enforcement of Data Integrity Constraints

There are essentially three types of data integrity constraints; in each case, its function is to put some limit on the value that a given data element can have. The first is field constraints, also known as edit constraints. Field constraints restrict the values allowable in a data element; for example, restricting the value that may be put into an "age" field to a number between 1 and 99. Intrarecord constraints limit values between elements in the same relation; e.g., if a sales region is Region Seven, all monthly sales reports serial numbers must begin with a 7. Interrecord constraints are like intrarecord except that they apply when the data elements are in different relations. [Kroenke, 1983, p. 179]

Enforcement of data integrity constraints may be active (e.g., embedded in code), "semi-active" or passive (e.g., data dictionary). These are discussed in the next two subsections.

a. Active Enforcement

Active enforcement occurs when user-written programs or data base management system (DBMS) routines are used to prevent the entry of data values which violate integrity constraints. For example, the use of edit templates such as the dBase III "picture" and "function" commands can restrict the length, character type, range, or format of a value provided by a user. Customized checking routines which only allow certain values are another form of active enforcement (e.g., a routine which will only allow a "P" or "M" for plant or minor property in the PROP_CODE data element). The use of temporary memory variables to store a user's input until it is error checked is also a form of active data integrity enforcement. The essential ingredient in each of the above cases is that program execution will not continue until the user inputs a permissible value, thus ensuring that inadmissible data does not corrupt the data base.

b. Semi-active and Passive Enforcement

Accuracy, consistency, and quality can also be supported by the use of a data dictionary in both a semi-active or passive role. An example of a semi-active approach is the placement into the data dictionary of upper and lower limits on the value of a data element. For example, if the value of MFG_YEAR were known to be between 1965 and 1987, a query such as the one shown in Figure 3.7 would isolate those records that contained invalid MFG_YEAR values.

```
.USE MFG_INFO
.SELECT 2
.USE DD_DELEM
.SELECT 1
.FIND MFG_YEAR
.DISPLAY MFG_NAME, NOUN_NAME FOR DD_DELEM->MFG_YEAR >
 MAX_YEAR .OR. DD_DELEM->MFG_YEAR < MIN_YEAR
```

Figure 3.7 Data Integrity Query Using the Data Dictionary

In addition to facilitating review of records for inadmissible values, the semi-active approach permits changes to the bounds of values without altering the program code--a feature that should appeal to data base administrators and programmers alike.

If the data dictionary is to be used passively, implementation is critical. Successful implementation of a data dictionary, whether automated or manual, has four characteristics: (1) its use and benefits are understood by those expected to use it, (2) it is readily accessible to users, (3) its use is enforced with minimal (preferably no) exception and (4) it has high credibility. Credibility requires that a data dictionary be regularly updated and internally consistent and reliable so that users will trust its contents.

In sum, data integrity constraints can be enforced actively, semi-actively, or passively. The active approach is likely to provide the most control and possibly the highest level of data integrity. However, it has the disadvantage of being least flexible and least tolerant of exceptions to the rule. At the opposite extreme is the passive avenue which relies heavily on user initiative. In most organizations a combination of approaches is used. Finally, it should be noted that while data integrity constraints can enhance data integrity, they also may increase vulnerability. If a constraint (e.g., a field constraint) can be enforced by embedding it in the code it can be an asset; however, if a constraint must rely on user memory or standard operating procedure (e.g., some interrecord constraints), it may be a liability. The following discussion of constraints in our code illustrates this point.

c. Data Integrity Enforcement in the PMPS

In the PMPS, active data integrity control mechanisms are used. The data dictionary has already been described. Some examples of data integrity constraints that are embedded in the code (field constraints) follow.

- Edit templates are interspersed throughout, especially in the code of format (".fmt") programs; see Appendices B and C for examples.
- Where possible in the structure definitions, data types of type "date" or "numeric" have been specified; dBase III will restrict input to data of those types.
- The first five digits of REPT_NO are hard coded into programs where called.
- "P" or "M" are the only values allowed in the PROP_CODE data element.
- Routines are used to check for duplicate NID numbers and stock numbers.

- GETNID.PRG checks for proper length, format, and numeric characters before accepting a NID number from the user.

As mentioned previously, in each of the above cases program execution will not continue until a valid entry is provided.

There are also two interrecord constraints worthy of note. The first is perhaps the most important. Since NID numbers are unique, the user must use the same NID_NO value in order to retrieve the correct data from more than one relation. This is not presently embedded in the code in all instances where a NID number is solicited from the user nor can it be. Thus in some cases it is incumbent upon the user to ensure that he or she has entered the correct NID number in order to retrieve data from several relations. An additional interrecord constraint exists between the MFG_INFO and ITEM_DES relations. In order to obtain a physical description of the item represented by any given NID number, the stock number in MFG_INFO must match a stock number in ITEM_DES. This matching is not embedded in the code and must rely on user care in data entry.

Finally, there are three field constraints which have yet to be exploited:

- By definition, if the value in TYPE_CODE is "1", then the property is always plant property (versus minor).

- If the first number in the STOCK_NO is a "7" then the equipment is some type of ADPE.

- The value in the field JULIAN should be restricted to no greater than 366.

All three of the above could provide an added element of control by enforcing data integrity.

2. Role of the Data Base Administrator

At the beginning of this section it was stated that there are two important ways to effect control in a data base environment. One is through data integrity constraints, many of which can be imposed by the designer or implementor of an automated support system. The responsibility for the other, the implementation of the Data Base Administrator (DBA) function, falls largely on the shoulders of the user. The next section briefly reviews highlights from the literature followed by relevant issues for MCD.

a. Review of the Literature

There is considerable variety in what authors say about the DBA. There seems to be no consensus on a standard set of duties and responsibilities. There is agreement, however, that the scope of his or her responsibilities should extend over all data in the data base, and by many definitions, over data outside the data base as well. There is also agreement that his or her general objective is to achieve a balance between controlling the quality and integrity of shared data and supporting users' needs. To accomplish this the DBA must at a minimum establish control over data definitions and data usage in all applications. [Perry, 1980, p. 29; Van Duyn, 1982, pp. 27-29].

In an environment with a large data base, automated systems, and many users who are sharing data, control over data definitions and data usage implies three areas of control: data activity, data base structure, and the data base management system. The kinds of activities that are included in each of these three areas are summarized in Figure 3.8.

As was discussed in Section III. C., one of the most effective tools at the disposal of the DBA is the data dictionary. The data dictionary is particularly important in the following functions: control of data elements, reduction of data redundancy and inconsistency, enforcement of standard definitions and usages, and the determination of the impact of data element changes on data base design.

b. Relevant issues for MCD

The preceding discussion has been basically generic. Often the DBA is a full time responsibility for one individual or even a dedicated staff. In a smaller operation like MCD, the need for a dedicated DBA is not as critical; however, the need for data base integrity is not diminished in the least! Any data base system, automated or not, is only as good as the quality of data in the data base.

MANAGEMENT OF DATA ACTIVITY

DBA must establish publish and enforce:

- **name/format standards for fields, records & files**
- **data ownership, access and modification rights**
- **backup and recovery procedures**

MANAGEMENT OF DATA BASE STRUCTURE

- **design and implement the data base schema**
- **control data redundancy**
- **maintain configuration control**
- **maintain documentation on data base structure**

DBMS MANAGEMENT

- **monitor system performance**
- **analyze and correct user complaints**
- **fine tune DBMS performance and design**
- **evaluate and implement improvements**

Figure 3.8 Responsibilities of the DBA

Consequently while an organization may not have a dedicated DBA, the preceding controls are still important.
For MCD this philosophy translates to the following kinds of controls (at a minimum).

The most crucial controls will be over the NID number. Procedures must be established and enforced which will control who may assign a NID number, when and how a number is assigned, and how to keep track of NID numbers previously assigned.

Of vital importance is the establishment of, and rigorous adherence to, backup and recovery procedures for information in the automated data base. A policy of weekly printouts of all new DD 1342 forms and daily backup of disks should be carried out at a minimum. Part of implementing this policy would be the designation of a single point of contact whose responsibility this would be.

At least one person should be sufficiently knowledgeable in dBase III to be able periodically to go into the data base and conduct "quality control by browsing around". This technique involves scanning through the data base often enough to be able to spot anomalies, correct them, and perform other "housekeeping" functions.

The data dictionary begun with this effort would be a good starting point for the establishment of a master data dictionary for MCD. Although time consuming, the conscientious maintenance of this document will more than pay for itself in problems avoided by virtue of improved quality of the data base. Depending on the assignment of responsibilities, it may be worthwhile to maintain a hard copy of the data dictionary so that microcomputer access is not a stumbling block for access to the data dictionary.

Finally, in the event that the automated system expands to a networked system with other offices, all of the above takes on a significantly increased new importance. At that time, it would be conceivable for MCD to require a full time, dedicated DBA.

IV. PROGRAMMING IMPLEMENTATION

A. REVIEW OF ROSS/SMITH DESIGN

The coding process, combined with frequent interaction with the user, verified that the Ross-Smith approach is essentially correct. There were however, two areas requiring notable modification. First, some of Ross-Smith's modules were revised to reflect the "real world" process more accurately. These changes make the structure of the system easier for the user to understand and easier for the programmer to translate data flows and functions into code. For example, the functions performed in the modules COMBINE 1342 INFORMATION and UPDATE FILES (See Ross-Smith, pp 180-181) are effectively performed simultaneously and do not need to be separate modules in the code. Other revisions were made to Ross-Smith hierarchy charts--especially in the acquisitions process. These changes came as a result of combining lower level functions into single modules (compare Figures 4.1 - 4.3 to Ross-Smith, pp. 161-163).

The second area of noteworthy change was discovered after further investigation into local use of form DD 1342. While Ross and Smith accurately describe the makeup of a DD 1342, several NPS-instituted modifications to the contents of the form are routinely made. They include:

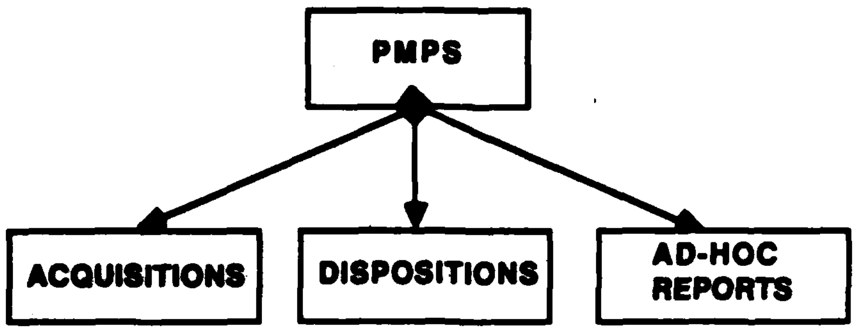


Figure 4.1 Top Level System Hierarchy Chart

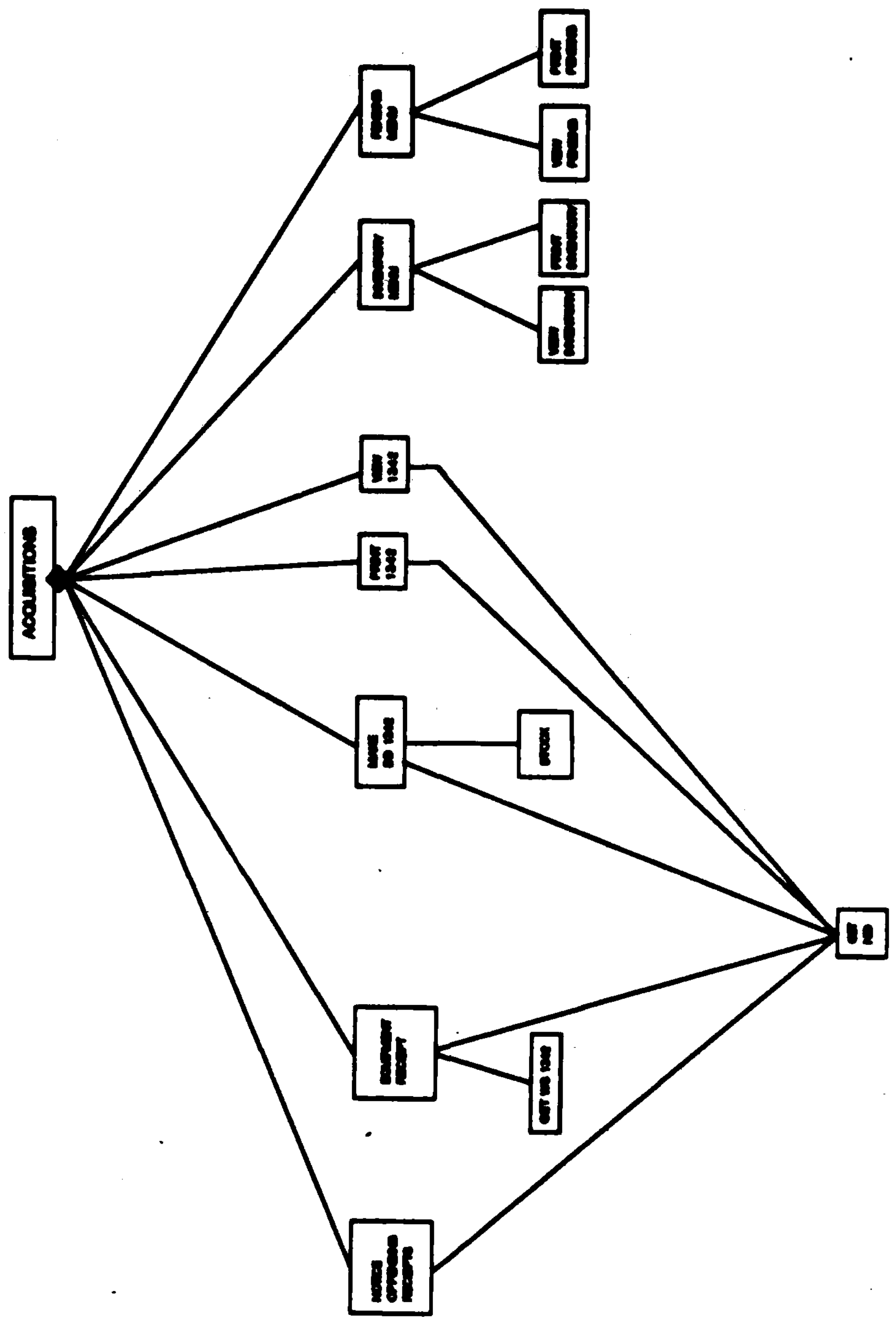


Figure 4.2 Acquisition Hierarchy Chart

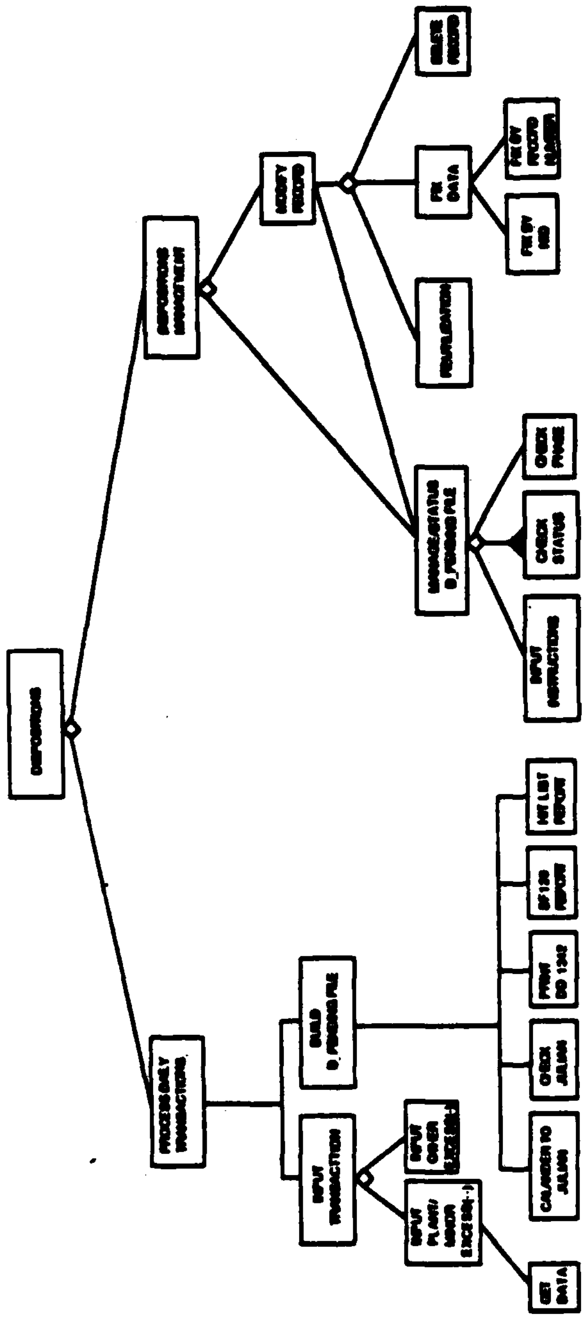


Figure 4.3 Disposition Hierarchy Chart

- Addition of the data element "PROP_CODE" next to NID number in block 3.
- Addition of the data element "DEPT_CODE" in the space beside the heading "Inventory Record".
- Use of the data element "PO_NO" in block 25 instead of the contract number as labelled.
- Addition of data elements "REQN_NO", "DATE_RECV," and "CONSIGNOR" to block 54.

Not all of these changes are presently operational in the code; those that are not are noted in software documentation. It should be stressed that these alterations to the Ross-Smith effort are not considered to be shortcomings. Rather they are a strong demonstration of the need for a prototyping methodology and regular interface with the user--even after a thorough requirements definition and system specification have been completed.

B. OBSERVATIONS ON THE CODING PROCESS

Two phenomena occurred during the coding process that are noteworthy. First, as expertise improved in the application language, better ways were discovered for programming modules that were thought to be completed. This placed the programmer in a quandary between periodically improving code already "finished" (slowing project momentum), or continuing development knowing that performance, error checking, or user interface could be significantly improved. This dilemma is endemic to the prototyping process and it is not uncommon eventually to have to re-code large portions of a project.

This should be expected due the nature of prototyping; however, knowing when to "retrench" is difficult. The ability to recognize this situation and establish guidelines for handling it would be very useful for managers who are in the position of managing or monitoring the progress of a programming effort.

Also worth noting is our experience confirming the "90% syndrome" [Boehm, 1981, Chapter 32, Section 4]. After a certain point in the coding, we found ourselves estimating code to be 90% complete only to discover that "completing" the remaining 10% uncovered an additional 10% which needed attention. It is noted here because it is felt that this syndrome can have a considerable detrimental effect on both programmer morale and project progress and thus should not be taken lightly by project managers.

C. USER INTERFACE

The purpose of this section is to provide a summary of what is contained in Appendices B-D. These appendices are significant in that they represent the stopping point in the development effort and therefore the starting point for any follow-on work.

All of the modules in the hierarchy charts in Figures 4.1 - 4.3 are coded and operational. The code for these modules is contained in Appendices B and C, with a User's Quick Reference in Appendix D. Additional performance, error check, and user interface enhancements should be made to some of the modules in Appendices B an C; recommendations are discussed in detail in Chapter V.

Not every coded program is discussed in the following; detailed information on all of the code is contained in the Appendices. Illustrations of the menus referred to below are provided in Appendix D.

PMPS.PRG - The program that provides the initial interface with the user is PMPS.PRG. It is the first module called when the user initiates the system. PMPS.PRG displays a simple menu that provides access to the Acquisition Menu, the Disposition Menu, Ad Hoc Queries or a return to the DOS prompt ("Quit"). The user may not input anything other than one of the four menu options. In each case (except "Quit") the selection results in another menu.

GETNID.PRG - Probably the simplest and yet one of the most important programs in the code, GETNID.PRG is used by other programs when a NID number input by the user is needed. Its function is to ensure that the user types the NID number in the correct format: the exact length and with only numeric values. It thus performs the crucial function of preventing common typing errors when inputing the NID number, which is the key data element used for virtually all operations. The GETRPT.PRG program is virtually identical except that it prompts the user for a report number used in the disposition process.

ACQUISIT.PRG - This program provides access to subprograms that parallel the NPS Supply Department's acquisition process and other primary MCD equipment control functions. It displays a menu with an option that corresponds to each of the major acquisition phases and the inventory process. ACQUISIT.PRG will loop until one of the menu options is selected. The first five of the options provide interaction with various data bases; the next two respond with other menus. The bottom option returns the user to the PMPS.PRG menu.

NT_PEN_R.PRG - The subprogram that parallels the first phase in the acquisition process, that is, the creation of a new logical record for a new piece of equipment, is NT_PEN_R.PRG (the module name is an abbreviation of Notice of Pending Receipt). This phase is initiated when MCD is first notified that a piece of equipment is on order or enroute. Often little is known at this point. However, once an automated system is in use, MCD standard operating procedure will require that at minimum the NID number must be assigned. Consequently the NID number and any other information that is available will be entered in this step of the process. Interface with the user begins with a prompt for the NID number (GETNID.PRG); the program then checks to ensure that the NID number input has not been assigned to another piece of equipment. If it has, an error message is displayed and the user is prompted to try again. Once a valid (i.e., new)

NID number is entered, the program presents the user with a series of three screens representing three relations: Pending, Property, and Purchase. The user is allowed to fill in fields that are known at that time and leave others blank. The information is then stored in each relation.

EQ_RECPT.PRG - The next program on the ACQUISIT.PRG menu is called after a new piece of equipment has been received but before it is sent to the destination department. As in NT_PEN_R.PRG, the first task is to ensure that the NID number entered is valid; however, this program ensures that there is a record in the logical Pending file corresponding to the NID number input, i.e., that the first phase (above) has been completed. Having found the NID number in the file, the program then checks to be sure that it has not already been through phase two (this program). This determination is made based on whether there is an 'R' (received) in the PEND_STAT field of the Pending relation.

Having determined that the user is in the right program for the right phase, the system prompts the user for a designation of 'P' or 'M' for plant or minor property. Finally, a sequence of three screens is displayed, one each from the Pending, Property, and Purchase data bases (relations). The user then inputs known information into the displayed fields.

MAKE1342.PRG - This program represents the third major phase of the acquisition process. Once a piece of equipment

has been sent to the destination department and the department has returned a form
WS 1342, MCD will fill in the remaining data fields in the data record for the new piece of equipment. Thus a DD 1342 Property Record Card can be completed. MAKE1342.PRG prompts the user for a NID number and then, in sequence, calls one screen each for the Plant and Inv_Info relations. There the user may input or change information as needed. The program then prompts the user for the appropriate equipment stock number; if that stock number is not already in the Item_Des relation, a screen is presented so the user may fill in a physical description of the equipment with that stock number.

VIEW1342.PRG - Representing the next logical step, VIEW1342.PRG allows the user to see the information that has been gathered in the last three steps, in
DD 1342 format. Called from the ACQUISIT.PRG menu, VIEW1342.PRG prompts the user for a NID number. The NID number then identifies the appropriate records in the Property, Plant, Mfg_Info, Inv_Info, and Purchase relations. The necessary data is extracted from each relation and displayed. Because of the length of the DD 1342, the screen display is broken into two screens and some unused parts of the DD 1342 are not shown at all.

PRNT1342.PRG - This program is also called from the Acquisition Menu; the program prompts the user for a NID number and then prints a form DD 1342 on blank paper using inputs from the same relations as VIEW1342.PRG. This program does not print onto a blank DD 1342 form, although this would be a desirable enhancement.

PENDMENU.PRG - Called from the Acquisition Menu, PENDMENU.PRG displays a menu. It allows the user three options: to browse or edit the logical Pending file (via VIEWPEND.PRG), to print all of the files that contain a certain value in the PEND_STAT field of the logical Pending file (via PRNTPEND.PRG), or to return to the Acquisition Menu. PENDMENU.PRG will not accept any input other than one of the menu options.

VIEWPEND.PRG - When the user selects the "Browse/Edit Pending Files" option of the menu displayed by PENDMENU.PRG, VIEWPEND.PRG displays selected portions of equipment records in the logical Pending file (the logical Pending file having been "built" from the Property, Purchase, and Pending relations). Which records are displayed is determined by the user; selection is based on the contents of the PEND_STAT field of the Pending relation. Once specified, the user will be stepped through the logical Pending file, one record per screen, until all records meeting the selection criteria have been displayed. Changes made on the screen are stored in the appropriate relation.

PRNTPEND.PRG - This program is also called from the PENDMENU.PRG Menu and allows the user to print out selected records in the logical Pending file. Its operation is conceptually the same as VIEWPEND.PRG except that the output is sent to the printer instead of the screen.

INVMENU.PRG - Called from the ACQUISIT.PRG Menu, this program displays a menu that will invoke either of two inventory functions or returns the user to the ACQUISIT.PRG Menu. The first inventory function is to browse or edit screens made up of DD 1342(-) information. The second menu option allows the printing of the same screens. INVMENU.PRG will not accept input other than one of the menu options.

VIEWINV.PRG - VIEWINV.PRG is called from INVMENU.PRG. It creates screens of DD 1342(-) information for equipment that meets two conditions input by the user. The conditions are inventory cutoff date and department of interest; this combination is the most common selection filter in the inventory process. VIEWINV.PRG uses information from the Property, Plant, Mfg_Info, and Inv_Info relations. Records are displayed, one screen per record, until all those meeting the selection criteria have been shown. Changes made on the screen are stored in the appropriate data base.

PRNTINV.PRG - When the user selects the "Print Inventory List" option from the Inventory Menu, this program is invoked. As with the VIEWPEND/PRNTPEND pair, PRINTINV.PRG sends output to the printer rather than the screen. It uses the same selection criteria and relations to provide an inventory list.

DISPOSIT.PRG - Upon selecting the Disposition portion of the automated system, the operator will get a screen allowing him or her to process daily transactions. This involves adding new records to the Pending Dispositions files or managing the existing records. The user may also print out the hit list report at this point.

INPUT_TR.PRG - If "Process Daily Transactions" is selected, the user will be asked if the excess equipment being processed has a NID number or not, as the processing of the equipment is somewhat different for the two situations.

INPT_EXC.PRG - If the excess equipment has a NID number, then screens requesting necessary information from the user will be displayed through this module. The information gathered here is used to build a "Pending disposition" record. As the user has stated there is a NID number for the equipment, most of the information is already resident in the data base. The information taken here is loaded into the data base and an "I" is inserted into the PEND_STAT field, indicating further processing is necessary.

OTHEREXC.PRG - If the user states that there is no NID number for this excess equipment, then the equipment is something other than plant or minor property. Although far less information is necessary to keep track of the Pending disposition in this case, none exists in the data base. Since the item does not have a NID number the system identifies the record by entering the current date and stock number in the NID_NO field. The information taken from the user will be loaded into the data base and an "I" is inserted into the PEND_STAT field to indicate further processing is necessary.

BLD_DPEN.PRG - Here the user is asked to validate the various screening periods; this is done since Navy regulations may change them in time. From that point on the system will process those records that have an "I" in the PEND_STAT field according to what type of equipment it is: Industrial Plant Equipment, Automated Data Processing Equipment or something else. This is determined from the information in the data base. At the conclusion, several reports are generated as necessary and or requested. They are:

- SF 120 Report
- DD 1342 Industrial Equipment Report
- Reutilization Hit List

DISP_MGT.PRG - Should the user select the Disposition Management option on the Dispositions Main Menu, the next screen allows another choice between "Managing the Status" of existing records, "Modifying Records" in the data base, or returning to the Main Menu.

MGTSTAT.PRG - "Managing Status" is selected when it is necessary to update a record or records or to process records for items that have reached their termination date. A menu is provided from which the user may select one of the above options.

INPUTIN.PRG - If the user has indicated that instructions from either DIPEC or DAREC have been received, then he or she is asked to enter those instructions for the record identified. The termination date and phase are adjusted in accordance with the instructions and the user is returned to the Disposition Management Menu.

MODREC.PRG - The user may want to modify the data base when selecting options from the Disposition Management Menu ("Modify Data Base"). The screen that appears allows the user to choose from several options, including those based on receiving a signed DD 1149, receiving a signed DD 1348-1, making corrections to data, printing a DD 1342, deleting a record, or processing an item for reutilization. When a signed DD 1149 or DD 1348-1 is received it is an indication that an item that has been transferred or disposed of has been received by the appropriate organization and now should be removed from the files.

DLET_REC.PRG - The user is asked if a NID number is available, corresponding to the record to be deleted. If one is, it is entered and the record is deleted throughout the data base. If the user does not have a NID number he or she is told to use the "Modify Record" option to browse the data base in order to find the record, get the NID number, and return to delete the record.

REUTILIZ.PRG - If an item is identified for reutilization at NPS, the NID number is entered, the user is prompted for the new department and location, and the system updates the appropriate relations. The record is then removed from the "Pending disposition" file. If the user is unsure of the NID number, the record number identifying that item as excess property may be entered. Since more than one item may be associated with one record number, the user will be allowed to browse each record with that number until the correct record is located.

FIX_DATA.PRG - When it is determined that there is an error within a record or set of records, there are two possible ways of getting to that record within the automated system in order to correct the error. The user may use either the NID number or the report number. Using the report number may require hunting through several other records to find the one needing the correction. All the data pertaining to the current record is displayed in a series of screens and the user may make any correction necessary.

Appendix D contains the User's Quick Reference Guide. This guide consists of reference sheets for use with each of the major modules of the system. Each follows the same format; an annotated sample of that format follows.

USER'S QUICK REFERENCE

1. PROGRAM NAME: The name of the program; that is, the name that is necessary to run the program.

2. PROMPT: Most modules will not have a prompt, per se, but rather are menu driven.

3. PURPOSE: A brief description of the program's primary function. How it interfaces with the user also is noted, with regard to what the system expects as input, what the output is, and its output form.

4. TO RUN: What is necessary to start this program. Most programs will be called by other programs and thus are not executed by the user directly.

5. TO USE: Particular information that is necessary for effective program use.

a. Listed here as subheadings, will be references to other pages having specific information about various operations within the system.

b. These references include the major subsystems and the various major operations within those subsystems.

6. TO GET OUT: What the user can do to escape from situations where he or she does not wish to be. This information is provided in general terms, allowing more detailed information to be covered in the more specific subsections of the TO USE section.

7. **PROBLEMS/CAVEATS:** "Nice to know" information that the user should keep in mind when running the program. These would include lessons learned by others who have used the system before. Many of the specifics here are also covered in a subsection of the TO USE section.

D. SAMPLE QUERIES AND RESPONSES

Chapter II contains ad hoc queries representative of the type expected for the automated system. The following are those queries repeated with a reference to a corresponding figure. Each figure contains the dBase III Plus "dot prompt" commands necessary to answer the query and screen print the response from an artificial data base. Any of these "dot prompt" sequences could be coded into the automated system and stored as a menu-driven recurring query or stored in a dBase II Plus "query file". For purposes of the demonstration, departments are designated by a four digit number (e.g., 1111, 2222, 3333).

- What plant/minor property does the Administrative Science Department hold? Figure 4.4.
- Where is (some particular) piece of plant/minor property located? Figure 4.5.
- What dollar value in minor property is held by the Electrical Engineering Department? Figure 4.6.
- Where is all of the XYZ Corp. equipment (e.g. for maintenance)? Figure 4.7.
- What dollar value in plant/minor property was lost when Root Hall burned to the ground? (Assumes

Root contains departments 1111 and 3333.) Figure 4.8.

- How big is a certain piece of plant property (e.g. for shipment)? Figure 4.9.

```
. USE PROPERTY
. DISPLAY NID_NO, NOUN_NAME, MFG_NAME FOR DEPT_CODE="4444"
Record#  NID_NO        NOUN_NAME                        MFG_NAME
      4  11111-111004 Offset Printing Press            Williams Paper
      8  11111-111008 Portable Viewgraph Projector     Alan's Video
     12  11111-111012 Computer workstation             Baker Furniture
```

Figure 4.4 dBase III Plus Response to Query about Administrative Science Equipment

```
. USE PROPERTY INDEX NID_PROP
. SELECT 2
. USE INV_INFO INDEX NID_INV
. SELECT PROPERTY
. SET RELATION TO NID_NO INTO INV_INFO
. DISPLAY NOUN_NAME, MFG_NAME, INV_INFO->LOCATE FOR NID_NO="11111-111013"
Record#  NOUN_NAME                             MFG_NAME             INV_INFO->LOCATE
     13  File cabinet, walnut                  Casavetti's Casement R100B211
```

Figure 4.5 dBase III Plus Response to Query about Location of a Piece of Property

```
. USE PROPERTY INDEX NID_PROP
. SELECT 2
. USE MFG_INFO INDEX NID_MFG
. SELECT PROPERTY
. SET RELATION TO NID_NO INTO MFG_INFO
. DISPLAY NOUN_NAME, MFG_INFO->COST FOR DEPT_CODE="2222"
Record#  NOUN_NAME                          MFG_INFO->COST
      2  Telescope                                 6543.00
      6  Atom Smasher, portable                    2496.00
     10  Disk Backup System                        7654.00
```

Figure 4.6 dBase III Plus Response to Query about Property Dollar Amount in Department Code "2222"

```
. USE PROPERTY INDEX NID_PROP
. SELECT 2
. USE MFG_INFO INDEX NID_MFG
. SELECT PROPERTY
. SET RELATION TO NID_NO INTO MFG_INFO
. DISPLAY NOUN_NAME, MFG_INFO->COST FOR DEPT_CODE="1111".OR.DEPT_CODE="3333"
Record#  NOUN_NAME                              MFG_INFO->COST
      1  Revolver, .32 cal, brk action                 554.00
      3  Wall Bookshelf, walnut                        678.00
      5  Automatic Fertilizer Spreader                  91.00
      7  Industrial Strength Labelmaker                567.00
      9  Mechanical Lift                              6594.00
     11  Half ton pick up truck                      19349.00
     13  File cabinet, walnut                          433.00
```

Figure 4.7 dBase III Plus Response to Query about Location of all "Baker Furniture" Equipment

```
. USE PROPERTY INDEX NID_PROP
. SELECT 2
. USE INV_INFO INDEX NID_INV
. SELECT PROPERTY
. SET RELATION TO NID_NO INTO INV_INFO
. DISP NOUN_NAME, MFG_MOD_NO, INV_INFO->LOCATE FOR MFG_NAME="Baker Furniture"
Record#  NOUN_NAME                         MFG_MOD_NO     INV_INFO->LOCATE
      3  Wall Bookshelf, walnut            Model 11454    R002B112
     12  Computer desk and shelf           WS-1234        R311B400
     13  File cabinet, walnut              FC-334-W       R100B211
```

Figure 4.8 dBase III Plus Response to Query about Dollar Value of Property Attached to Departments with Code Numbers "1111" or "3333"

```
. USE PLANT
. DISPLAY NID_NO, LENGTH, WIDTH, HEIGHT, WEIGHT FOR NID_NO="11111-111007"
Record#  NID_NO        LENGTH WIDTH HEIGHT WEIGHT
      7  11111-111007 25in   19in  12in   199lb
```

Figure 4.9 dBase III Plus Response to Query about the Size of a Piece of Equipment

V. THE TASK REMAINING

A. THE ORGANIZATIONAL ENVIRONMENT AS OF AUGUST 1987

1. Personnel

Currently MCD has a new division head and there is one experienced worker each in the acquisition section and disposition section. Additionally, a new employee has just reported to support the acquisition process, specifically for the additional workload created by the new responsibility of accounting for and control of minor property. Advertising for the positions of joint acquisitions/dispositions supervisor and for an additional person in the disposition section is being delayed due to a lack of funding.

The new acquisition section employee has experience with mainframe computers and is eager to learn to use microcomputers and the proposed automated system.

2. Equipment

The MCD has ordered a microcomputer which is to be dedicated to the new automated system. The machine ordered is a Zenith model ZFX-248-GE with the following characteristics: IBM AT compatible, one 5 1/4-inch 360KB floppy disk drive, one 20MB hard disk drive, 80286 8MHZ CPU card with 512KB RAM, input/output card with serial and parallel ports, enhanced graphics adapter, dual frequency high resolution monitor, and 640KB memory expansion board.

The MCD also has a Diablo daisy wheel printer that will be shared with MCD personnel not using the automated system.

B. TASKS FOR THE SUPPLY DEPARTMENT USERS

The single most important task for the Supply Department user will be to maintain the development momentum. The arrival of new computer hardware in five months will contribute to this effort; until then it might be possible to borrow appropriate computers from another department. The equipment that is currently in the MCD office is unacceptable except for the most basic testing and familiarization tasks. Key personnel to help keep the project moving include the Computer Technology Curricular Officer and Academic Associate for the Information Systems Curriculum (both of whom may assist in identifying potential student/faculty programmers to continue development), and those in the military chain of command (who provide funding and hiring approval). Other necessary tasks include:

1. Training. There are many commercial training programs available for the operation of dBase III Plus. It is <u>highly</u> recommended that at least one MCD individual be sent to one of these courses (two to three days minimum).

2. Printer. The identification of a dedicated printer for the automated system is desirable. Follow-on programmers should be able to provide specific details on technical characteristics that will best suit the system in its final form.

3. dBase III Plus. MCD should purchase its own copy of dBase III Plus software and documentation to have resident in division spaces; this is

particularly important if a staff member is going to be sent to school.

4. Operating Procedures. It is important that the MCD understand some basic tenets of database integrity and database administration, and incorporate these into the office standard operating procedures. The discussion in Chapter 3 will provide a foundation for development of these procedures.

5. System Performance. Users of the automated system should continuously monitor its performance, particularly as the data base grows in size. Although Dbase III Plus should be capable of supporting expected MCD operations for the present, unexpected growth in data base size or increased complexity of applications could slow performance to an unacceptible level.

C. TASKS FOR FOLLOW-ON DEVELOPERS

Tasks for follow-on programmers may be divided into three catagories: support documentation, lingering research issues, and software code.

1. Support Documentation

At a minimum, a full and detailed User's Manual must be developed before the system is ready for any extended use by the user. An on-line tutorial would also be helpful.

2. Research

At least three areas need further research before development progresses very much further. The first is system sizing; although the data base probably will not contain as many as 65,000 items, nothing more specific is available at this time. Recognizing that management decree has a significant impact on sizing, the follow-on developer may have to press for some decisions in order to continue development without a high degree of risk.

The second area of research concerns expansion of the system: a logical outgrowth of an automated accounting system is the expansion to the NPS departments via a network. This will have considerable impact on design and development and details must be known at the earliest opportunity.

Finally, system and data base security is an area that needs further requirements definition. dBase III Plus can effectively support password protection; however, good security will require standard operating procedures as well. In any case, security issues will have an impact on development direction and emphasis.

3. Code

Specific areas where improvements are needed include error checking, system performance, and the user interface.

a. Error Checking

The following improvements will assist in reducing the most common errors expected with the automated system as designed:

- Within the automated system there should be a transaction record file and a mechanism with which to store records input by the user during an individual session. This file could then be printed out at the completion of the session and compared to what the user thought he or she had done. Upon finding an error, a correction could be made immediately.

- The first five digits of the NID number are always the NPS Unit Identification Code--62271; these should be "hard-coded" to preclude typographical errors.

- In the few remaining instances where input from the screen is obtained with the "accept" command, this should be replaced with an "@ GET" sequence so that "picture" and "function" capabilities may be used for error checking.

- In all cases where the user currently has direct access to all data bases (e.g., via an "edit" command), access should be facilitated via a memory variable that can be checked before updating the data base. Prague and Hammitt use a good approach in the referenced text. [Prague, 1986, p. 238]

- In the INPUT_TR module it is conceivable that the user might want to answer "yes" to the question posed, hit the wrong key by mistake, and end up in OTHEREXC and not realize it. There should be some text announcing where the user is and a mechanism that allows escape. The text and escape mechanism should be put in both INPT_EXC and OTHEREXC.

- In MGTSTAT, some textual information is needed indicating to the user where he or she is and a way to back up if the wrong key has been depressed.

b. System Performance

In the area of system performance, probably the most dramatic improvements can be made by use of a compiler. Several commercial packages are available for use with dBase III Plus including CLIPPER, QUICKSILVER, and FOXBASE PLUS. These packages claim similar benefits, such as faster execution time (10-100 times faster), enhanced security of code (by virtue of conversion to machine language), and increase in allowable memory variables and data base fields. While these claims have not been tested by the authors, it is clear that the use of a compiler would improve system performance enough to make use of one worthwhile.

Using a compiler such as CLIPPER has an additional significant benefit. As the code is currently written, occasional "joins" are used to create new data bases. As this actually physically creates a new database, it is both slow and uses system memory. With CLIPPER or a similar compiler, the command "set relation to" can be used to link up to ten data bases on a key field. This technique precludes the creation of a new data base, thus reducing additional memory needed and dramatically speeding up operation.

Another technique that will result in greatly increased speed is the use of "find" with a ".ndx" file. Using "find", the system searches on an index to find a record this is much faster than going through each record individually. In the original version of the code, records were located using the "locate" command. In most cases this has been replaced with a "find" command and an index. This should be done throughout the code.

The physical description (PHYS_DESC) field of Item_Des relation could be put into a "memo" type of field if the "joins" are eliminated (dBase will not allow memo fields to be "joined"). This would save memory and improve speed.

In the HTLSTREP module, the PEND-D and PROPERTY relations are joined in order to get all the information necessary to make the Hit List Report. This might be done better by setting a relation between the two and customizing the report. The report form is currently being used and does not allow for the use of two files concurrently.

The REUTILIZ module could be improved by setting the relations with the proper relationships up front, before any other code. In that way the system can access any of the information it may need. Also, messages are needed along the way indicating to the user what is taking place and a way to escape if necessary.

Finally, two general comments about system performance are appropriate. First, some programs do not observe programming guidelines that would result in low coupling and strong cohesion. Generally this is due to the lack of programming experience on the part of the implementors, and should be reviewed. Second, throughout the system there are modules that require the setting of relationships between the majority of the relations that make up the data base. Code could be read easier if this task were incorporated into a procedure and called when it was necessary.

c. User Interface

There are many ways to improve the user interface; some of the more significant ones include:

- As currently coded, the user cannot reverse direction when in the edit/browse option of the Inventory Menu or Pending Menu.

- Similarly, there is no way to escape gracefully when in the above mentioned modules; the user must go through all of the files to continue.

- If the user would support a reduction in the number of fields in the Pending and Inventory reports, they could be changed into "xxx.rpt" files and include several pieces of equipment per page, versus one per page as now coded.

- The manufacturer's name field (MFG_NAME) has been reduced from 30 to 20 characters in order to fit on a DD 1342; a second line of MFG_NAME could be added in order to bring the field length back to 30.

- In the BLD_DPEN module, if a piece of equipment is TYPE_CODE "1" it is classified as (IPE). Each excess IPE item must have an Idle Report sent to DIPEC. The Idle Report is simply a DD 1342 with the dispositions section filled in at the bottom. At the moment the DD 1342 information is printed out on a blank piece of paper. It would be more convenient to have it print out on a blank DD 1342 (at least the top portion of this document); the user could then fill in the rest to create the Idle Report.

- Also within the BLD_DPEN module, items identified as having a 7000 series stock number are ADPE. They are written up on a SF-120 which is sent to DAREC. The necessary information for each appropriate record is currently stored in a temporary file and later printed out in a report format from which a SF-120 can be typed. It would be far more convenient to have the SF-120 typed automatically after first instructing the user to mount the form on the printer.

- There have been some recent changes in the format of the form DD 1348-1, with slight variations from one disposal organization to another. It may be necessary to make different modules, one for each form, and allow the user to select the proper format from a menu.

D. CONCLUSIONS

The single overlying theme of this effort has been to answer the following questions: will this system designed by Ross and Smith work ? If so, will it support the NPS Supply Department adequately? And if this is so, is it worth pursuing? The authors believe that the answer to all three questions is resoundingly affirmative!

From the sizing standpoint, the immediate problem has been somewhat overcome by events. By management decree, some items that could technically be categorized as minor property will not be controlled and accounted for centrally (cost versus benefit). Additionally, also by management decree, only new equipment will be entered initially and existing equipment will be added on an as-time-permits basis. Consequently, the system data base size is largely unknown, but it is reasonably certain that it will be smaller than originally estimated and will grow slowly enough to allow planning for future capacity expansion. Current requirements can be handled by the system as designed with equipment on order.

Most data redundancy, insertion and deletion anomalies, and other similar problems have been eliminated with the degree of normalization applied to the original Ross-Smith file structure design. This contributes significantly to data base integrity.

System performance as currently coded has not been maximized; however, the basic structure of the code is sound. With the use of a compiler and some of the techniques mentioned above ("find" versus "locate", "set relation to" versus "join") system performance will be adequate for the foreseeable future.

dBase III Plus contains numerous powerful error checking aids such as the "picture" command (an edit mask) and other functions that support the use of memory variables, etc. These, combined with user interface screens coded with prompts, reminders, and helps, should provide the level of error checking necessary for operation at the clerk-typist level.

If properly used and maintained, the on-line data dictionary will provide support to both the user and the follow-on developer. This tool will provide a useful point-in-time reference as well as support for further development, for learning the system, and for maintaining the system and system documentation.

Finally, the system as designed and the code as written should be transportable to other U. S. Navy commands with minimal modification. Only a few items in the code are NPS unique (e.g. local modifications to the form DD 1342). Navy commands are all governed by the same NAVCOMPT manual and have all been directed to centralize the accounting of minor property. Consequently the structure and code should support a generic Navy property management operation.

List of References

Berzins, V., and Berzins, L., "Rapid Prototyping of Real-Time Systems", to appear in IEEE Transactions on Software Engineering, 1987.

Boehm, B. W., Software Engineering Economics, Prentice-Hall, Inc., 1981.

Dearnly, P. A., and Mayhew, P. J., "In favor of System Prototypes and Their Integration into the System Development Cycle", The Computer Journal, v. 26, February 1983.

Harrison, R, "Prototyping and Systems Development Life Cycles", Journal of Systems Management, v. 36, September 1985.

Kroenke, D. M., Database Processing: Fundamentals, Design, Implementation, 2d ed., Science Research Associates, Inc. 1983.

Department of the Navy, NAVCOMPT Manual, Vol. 3, Chapter 6.

Perry, W. E., "No. 10, Control in a Data Base Environment", Data Base Management, the QED Monograph Series, 1980.

Prague, C. N., and Hammitt, J. E., Programming with dBase III Plus, Tab Books Inc., 1986.

Personal Interview with LT. R. M. Hausvik, Head of Material Control Division, NPS Supply Department, 13 April 1987.

Ross, M. L. and Smith, R. Q., A Systems Analysis and Design Proposal for the Supply Department's Minor and Plant Property Process at The Naval Postgraduate School, Masters Thesis, Naval Postgraduate School, Monterey, California, March 1987.

Ross, R. G., Data Dictionaries and Data Administration, AMACOM, 1981.

Van Duyn, J., Developing a Data Dictionary System, Prentice Hall Inc., 1982.

APPENDIX A

DATA DICTIONARY SOFTWARE CODE

The following contains the software code contained in the Data Dictionary program DATADICT.PRG. Each module begins with a documentaion header. This header contains the name of the author, the date that the code was last modified, and a general description of the purpose of the module. Additionally listed are all of the programs, formats, and data base files (.PRG, .FMT, .DBF respectively) that the module uses, calls, or is called by.

```
*DATADICT.PRG
*Author:        B. A. Whitehouse
*               25 June 1987
*Purpose:       To allow the user to select what part of
*               the data dictionary he/she cares to display.
*I/O Files:     DD_FILES.DBF, DD_DELEM.DBF, DDPRGRM.DBF,
*               DD_CONTAN.DBF, DD_FILES.FMT, DD_DELEM.FMT,
*               DD_PRGRM.FMT, DD_CONTAN.FMT
*Called By:     None
*Calls:         DICFILE.PRG, ELEMENTS.PRG, DICPRGRM.PEG,
*               PGMFILES.PRG, ADDTODIC.PRG

clear
set talk off
store "  " to dic_choice
text

                    PMPS DATA DICTIONARY

The following screen will allow you to select the portion of the
data dictionary at which you care to look.  Files, lists the
relations used with in the PMPS.  Data elements, are the data
elements which are used in the part of the system which is
currently in operation.  Programs, gives a brief description of
the programs which make up the PMPS.  Programs / Files cross
reference, lists the files which are called by the various
programs.

endtext
wait to cont
clear
do while .t.
    @ 2,0 to 21,79 double
    @ 3,28 say [Data Dictionary Menu]
    @ 4,1 to 4,78 double
    @  7,23 say [Task Code      Dictionary File]
    @  8,23 say [    1           Files]
    @  9,23 say [    2           Data Elements]
    @ 10,23 say [    3           Programs]
    @ 11,23 say [    4           Files / Program Cross Reference]
  @ 12,23 say [    5           Append New Entries]
    @ 14,23 say [    6           Quit]
?
?
?
    wait "      Enter your choice (Task Code):  " to dic_choice

    do case
        case dic_choice = '1'
            do dicfile

        case dic_choice = '2'
```

```
                do elements

            case dic_choice = '3'
                do dicprgrm

            case dic_choice = '4'
                do pgmfiles

            case dic_choice = '5'
                do addtodic

            case dic_choice = '6'
                clear
                return

            otherwise
                loop

        endcase
    enddo

    *eof datadict.prg
```

```
*DICFILE.PRG
*Author:       B. A. Whitehouse
*Date:         25 June 1987
*Purpose:      This program allows the user to review the files
*              data dictionary.
*I/O Files:    DD_FILES.DBF, DD_FILES.SCR, DD_FILES.FMT,
*              DD_FILES.DBT
*Called By:    DATADICT.PRG
*Calls:        None

clear

f_name = space(25)
@ 10,5 say "Enter file name:" get f_name
read
use dd_files
set format to dd_files
edit for FILE_NAME = f_name

close all
clear
return

*eof dicfile.prg
```

```
*DICFILE.FMT
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       To create a screen for inputting changes to the
*               DD_FILES.DBF.
*I/O:           FILE_NAME, DESCRIPTION, CONTENTS, USED_BY
*Called By:     DICFILE.PRG, ADDFILES.PRG
*Calls:         None

@  1, 19  SAY "DATA DICTIONARY FOR FILES/RELATIONS"
@  3,  0  SAY "FILE_NAME"
@  3, 12  GET  DD_FILES->FILE_NAME
@  5, 43  SAY "**************************************"
@  6, 43  SAY "*  To view a MEMO press              *"
@  7,  0  SAY "File Description:"
@  7, 22  GET  DD_FILES->DESCRIPTN
@  7, 43  SAY "*    <Ctrl><PgDn>, to return         *"
@  8, 43  SAY "*    from MEMO press <Ctrl><PgDn>.  *"
@  9, 43  SAY "*                                    *"
@ 10, 43  SAY "*  Press <Esc> for previous menu.   *"
@ 11,  0  SAY "Fields contained in file:"
@ 11, 28  GET  DD_FILES->CONTENTS
@ 11, 43  SAY "*                                    *"
@ 12, 43  SAY "*  Press <PgDn> for next filename   *"
@ 13, 43  SAY "**************************************"
@ 15,  0  SAY "This file is used by:"
@ 15, 24  GET  DD_FILES->USED_BY

*eof DD_FILES.FMT
```

```
*ELEMENTS.PRG
*Author:       B. A. Whitehouse
*Date:         25 June 1987
*Purpose:      This program allows the user to review the
*              the elements of the data dictionary.
*I/O Files:    DD_DELEM.DBF, DD_DELEM.SCR, DD_DELEM.FMT,
*              DD_DELEM.DBT
*Called By:    DATADICT.PRG
*Calls:        None

clear

n_name = space(30)
@ 10,5 say "Enter nickname in all CAPS:" get n_name
read
use dd_delem
set format to dd_delem
edit for NICKNAME = n_name

close all
clear
return

*eof elements.prg
```

```
*DD_DELEM.FMT
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       To make input and changes to the DD_DELEM.DBF.
*I/O:           NICKNAME, FULLNAME, DEFINITION, FORMAT, USED_IN
*Called By:     ELEMENTS.PRG, ADDELEM.PRG
*Calls:         None

@  1, 22  SAY "DATA ELEMENTS DATA DICTIONARY"
@  5,  0  SAY "Nickname:"
@  5, 12  GET  DD_DELEM->NICKNAME
@  8,  0  SAY "Full Name:"
@  8, 12  GET  DD_DELEM->FULLNAME
@ 11,  0  SAY "Definition:"
@ 11, 12  GET  DD_DELEM->DEFINITION
@ 11, 20  SAY "To display MEMO press <Ctrl><PgDn>,"
@ 12, 26  SAY "Press <Esc> to return."
@ 14,  0  SAY "Format:"
@ 14, 12  GET  DD_DELEM->FORMAT
@ 17,  0  SAY "Used in:"
@ 17, 12  GET  DD_DELEM->USED_IN

*eof DD_DELEM.FMT
```

```
*DICPRGRM.PRG
*Author:        B. A. Whitehouse
*Purpose:       This program allows the user to view the data
*               dictionary listing the programs and their
*               outlined purpose.
*I/O Files:     DD_PRGRM.DBF, DD_PRGRM.FMT, DD_PRGRM.SCR
*               DD_PRGRM.DBT
*Called By:     DATADICT.PRG
*Calls:          None

clear

p_name = space(8)
@ 10,5 say "Enter program name in all CAPS:" get p_name
read
use dd_prgrm index pgm_name
select 2
use call_pgm index namecall
select 1
set relation to pgmname into call_pgm

set format to dd_prgrm
edit for PGMNAME = p_name

clear
return

*eof dicprgrm.prg
```

```
*DD_PRGRM.FMT
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       To make a screen with which to make changes and
*               input data to DD_PRGRM.DBF.
*I/O:           PGMNAME, FULLNAME, PURPOSE, CALLS, CALLEDBY
*Called By:     DICPRGRM.PRG, ADDPGM.PRG
*Calls:         None


●  1, 24  SAY "DATA DICTIONARY OF PROGRAMS"
●  4,  0  SAY "Program name:"
●  4, 14  GET  DD_PRGRM->PGMNAME
●  4, 27  SAY "(All program names end in .PRG)"
●  7,  0  SAY "The module name is"
●  7, 19  GET  DD_PRGRM->FULLNAME
● 10,  0  SAY "The purpose of this program is"
● 10, 32  GET  DD_PRGRM->PURPOSE
● 10, 39  SAY "To display press <Ctrl><PgDn>,"
● 11, 46  SAY "<Esc> to return."
● 13,  0  SAY "This program calls:"
● 14, 12  GET  CALL_PGM->CALLS
● 16,  0  SAY "This program is called by:"
● 17, 12  GET  CALL_PGM->CALLEDBY

*eof DD_PRGRG.FMT
```

```
*PGMFILES.PRG
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       This programs allows the user to view the
*               file / program cross reference file.
*I/O Files:     DD_CONTA.DBF, DD_CONTA.FMT, DD_CONTA.SCR
*Called By:     DATADICT.PRG
*Calls:         None

clear

p_name = space(8)
@ 10,5 say "Enter program name in all CAPS:" get p_name
read
use dd_conta
set format to dd_conta
edit for PGMNAME = p_name

clear
return

*eof PGMFILES.PRG
```

```
*DD_CONTA.FMT
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       To create a screen with which to make changes
*               and input to the DD_CONTA.DBF.
*I/O:           PGMNAME, USES
*Called By:     PGMFILES.PRG

@  2, 20  SAY "PROGRAM / FILE CROSS REFERENCE"
@  5,  0  SAY "Program name:"
@  5, 15  GET  DD_CONTA->PGMNAME
@  7,  0  SAY "Files this program uses:"
@  7, 25  GET  DD_CONTA->USES  FUNCTION "S54"  PICTURE
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"

*eof DD_CONTA.FMT
```

```
*ADDTODIC.PRG
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       To allow the user to select some portion of the
*               data dictionary to which they wish to append
*               to append additional entries.
*I/O Files:     DD_FILES.DBF, DD_DELEM.DBF, DDPRGRM.DBF,
*               DD_CONTAN.DBF
*Called By:     DATADICT.PRG
*Calls:         ADDFILES.PRG, ADDELEM.PRG, ADDPGM.PRG,
*               ADPGMFIL.PRG

clear
set talk off
store "  " to dic_choice
text

                    PMPS DATA DICTIONARY

The following screen will allow you to select the portion of the
data dictionary to which you care to append new information.
Files, lists the relations used with in the PMPS.  Data
elements, are the data elements which are used in the part of
the system which is currently in operation.  Programs, gives a brief
description of the programs which make up the acquisition portion of
the system.  Programs / Files cross reference, lists the files which
are called by the various programs.

endtext
wait to cont
clear
do while .t.
   @ 2,0 to 18,79 double
   @ 3,13 say [Appending New Information to Data Dictionary
Menu]
   @ 4,1 to 4,78 double
   @  7,23 say [Task Code      Dictionary File]
   @  8,23 say [    1           Files]
   @  9,23 say [    2           Data Elements]
   @ 10,23 say [    3           Programs]
   @ 11,23 say [    4           Files / Program Cross
Reference]
   @ 13,23 say [    5           Quit]
?
?
?
   wait "     Enter your choice (Task Code):  " to dic_choice

   do case
      case dic_choice = '1'
         do addfiles
```

```
          case dic_choice = '2'
              do addelem

          case dic_choice = '3'
              do addpgm

          case dic_choice = '4'
              do adpgmfil

          case dic_choice = '5'
              clear
              return

          otherwise
              loop

      endcase
  enddo

  *eof ADDTODIC.PRG
```

```
*ADDFILES.PRG
*Author:      B. A. Whitehouse
*Date:        25 June 1987
*Purpose:     This program allows the user to append new entries
*             to the files data dictionary.
*I/O Files:   DD_FILES.DBF, DD_FILES.SCR, DD_FILES.FMT,
*             DD_FILES.DBT
*Called By:   ADDTODIC.PRG
*Calls:       None

clear

f_name = space(25)
@ 10,5 say "Enter file name:" get f_name
@ 13,7 say "CHECK YOUR ENTRY!"
read
use dd_files
append blank
replace file_name with f_name
set format to dd_files
edit for file_name = f_name

close all
clear
return

*eof ADDFILES.PRG
```

```
*ADDELEM.PRG
*Author:      B. A. Whithehouse
*Date:        25 June 1987
*Purpose:     This programs allows the user to append
*             additional elements to the data dictionary.
*I/O Files:   DD_DELEM.DBF, DD_DELEM.SCR, DD_DELEM.FMT,
*             DD_DELEM.DBT
*Called By:   ADDTODIC.PRG
*Calls:       None

clear

n_name = space(30)
full_n = space(45)
form   = space(55)
where  = space(65)

@ 10,5 say "Enter nickname of data element in all CAPS :"
@ 11,10 get n_name
@ 13,5 say "Enter full name (lower case):"
@ 14,10 get full_n
@ 16,5 say "Enter format:" get form
@ 18,5 say "Enter where the element is used:"
@ 19,10 get where
read
use dd_delem
append blank
replace nickname with n_name
replace fullname with full_n
replace format with form
replace used_in with where
set format to dd_delem
edit for NICKNAME = n_name

close all
clear
return

*eof ADDELEM.PRG
```

```
*ADDPGM.PRG
*Author:     B. A. Whitehouse
*Date:       25 June 1987
*Purpose:    This program allows the user to append new entries
*            to the portion of the data dictionary listing the
*            dictionary listing the programs and their purpose.
*I/O Files:  DD_PRGRM.DBF, DD_PRGRM.FMT, DD_PRGRM.SCR
*            DD_PRGRM.DBT
*Called By:  ADDTODIC.PRG
*Calls:      None

clear

ans    = 'Y'
p_name = space(8)
full_n = space(30)
tocall = space(60)
called_by = space(45)

@ 5,5 say "Enter program name in all CAPS:" get p_name
@ 7,5 say "Enter progams full name:"
@ 8,10 get full_n
@ 10,5 say "Enter what programs it calls:"
@ 11,10 get tocall
@ 13,5 say "Enter what programs it is called by:"
@ 14,10 get called_by

read
use dd_prgrm index pgm_name
append blank
select 2
use call_pgm index namecall
append blank
select 1
replace pgmname with p_name
replace fullname with full_n
select 2
replace pgmname with p_name
replace calls with tocall
replace calledby with called_by
set format to dd_prgrm
edit for PGMNAME = p_name
clear

close all
clear
return

*eof ADDPGM.PRG
^Z
```

```
*ADPGMFIL.PRG
*Author:        B. A. Whitehouse
*Date:          25 June 1987
*Purpose:       This programs allows the user to append new
*               entries to the file / program cross reference
*               file.
*I/O Files:     DD_CONTA.DBF, DD_CONTA.FMT, DD_CONTA.SCR
*Called By:     ADDTODIC.PRG
*Calls:         None

clear
ans = 'Y'
use dd_conta
do while ans = 'y' .or. ans = 'Y'
    clear
    p_name = space(8)
    filename = space(12)
    @ 10,5 say "Enter program name in all CAPS:" get p_name
    @ 12,5 say "Enter what file the program uses:" get filename
    @ 14,10 say "There is one record for each file a program uses."
    @ 15,10 say "Use entire file name to include .prg, dbf, etc."
 @ 18,7 say "Do you want to enter another (Y/N)." get ans
    read
    append blank
    replace pgmname with p_name
    replace uses with filename
enddo

close all
clear
return

*eof ADPGMFIL.PRG
```

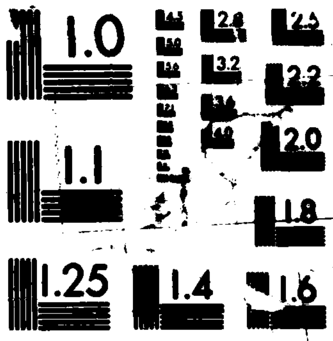
APPENDIX B

ACQUISITIONS MODULES SOFTWARE CODE

The following contains the software code contained in the modules that support the equipment acquisition process at MCD. Each module begins with a documentaion header. This header contains the name of the author, the date that the code was last modified, and a general description of the purpose of the module. Additionally listed are all of the programs, formats, and data base files (.PRG, .FMT, .DBF respectively) that the module uses, calls, or is called by.

AD-A187 783 THE IMPLEMENTATION OF A PLANT AND MINOR PROPERTY ACCOUNTING SYSTEM FOR THE NAVAL POSTGRADUATE SCHOOL(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W T KEY ET AL. 2/3

UNCLASSIFIED SEP 87 F/G 5/1 NL



1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

```
*PMPS.PRG
*Author:      W. T. Key
*Date:        17 July 1987 (Mod. 9/01/87 BAW)
*Purpose:     This is the program that starts it all; it
*             presents the user with a menu which allows
*             entry into the Acquisition, Disposition or
*             Ad-Hoc modules.
*I/O:         None (all .scr, .dbf, .fmt, etc go here)
*Called By:   None (appropriate .prg go here)
*Calls:       ACQUISIT.PRG, DISPOSIT.PRG, AD_HOC.PRG


clear
set talk off
set status off
store " " to choice

do while .t.
    @2,0 to 21,79 double
    @3,33 say [Main Menu]
    @4,1 to 4,78 double
    @6,20 say [Task                                 Task Code]
    @8,20 say [Acquisitions                                 1]
    @9,20 say [Process Disposition Transactions             2]
    @10,20 say [Disposition Management                       3]
    @11,20 say [Ad Hoc Requests                              4]
    @13,20 say [Quit                                         5]

    @15,15 say [Enter Task Code:] get choice
    read

    do case
        case choice = '1'
            do acquisit.prg
        case choice = '2'
            do proc_trn.prg
        case choice = '3'
            do disp_mgt
        case choice = '4'
            do ad_hoc.prg
        case choice = '5'
            clear
            exit
        otherwise
            loop
    endcase
enddo

set status on

*Eof  PMPS.PRG
```

```
*ACQUISIT.PRG
*Author:      W. T. Key
*Date:        17 July 1987
*Purpose:     Provides a control structure for access to the
*             various sub-modules of the acquistion process.
*             Access to each is via menu presented in this
*             module.
*I/O:         None
*Called By:   PMPS.PRG
*Calls:       NT_PEN_R.PRG, EQ_RECPT.PRG, MAKE1342.PRG,
*             PRINT1342.PRG, VIEW1342.PRG, INV_MENU.PRG,
*             PENDMENU.PRG

clear
store " " to choice

do while .t.
    @2,25 say [ACQUISITIONS MAIN MENU]
    @4,23 say [Task                    Task Code]
    @6,23 say [Notice of Pending]
    @7,23 say [Receipt                      1]
    @9,23 say [Equipment Receipt            2]
    @11,23 say [Reconcile 1342               3]
    @13,23 say [Print 1342                   4]
    @15,23 say [Modify 1342                  5]
    @17,23 say [Inventory Files              6]
    @19,23 say [Pending Files                7]
    @21,23 say [Return to Main Menu          8]
    @23,15 say [Enter Task Code:] get choice
    read

    do case
         case choice = '1'
             do nt_pen_r.prg

         case choice = '2'
             do eq_recpt.prg

         case choice = '3'
             do make1342.prg

         case choice = '4'
             do prnt1342.prg

         case choice = '5'
             do view1342.prg

         case choice = '6'
             do invmenu.prg

         case choice = '7'
             do pendmenu.prg
         case choice = '8'
             clear
             return
         otherwise
             loop
   endcase
enddo

*Eof  ACQUISIT.PRG
```

```
*NT_PEN_R.PRG
*Author:      W. T. Key
*Date:        09 August 1987 (Mod 9/1/87 BAW)
*Purpose:     Initial data regarding equipment which has been
*             ordered is entered into the database.  This
*             module is the entry point of a new piece of
*             equipment.  It is the beginning of a WS1342.
*I/O:         PENDING.DBF, PROPERTY.DBF, PURCHASE.DBF,
*             PENDING.FMT, PROPERTY.FMT, PURCHASE.DBF
*Called By:   ACQUISIT.PRG (PMPS.PRG)
*Calls:       GETNID.PRG


clear
set talk off
store .t. to check
text

**********************************************************

     This module lets you enter data on a new piece of
     equipment which has not been received yet.  You will
     be entering  data on the next four screens in order to
     start building your WS 1342.


**********************************************************

endtext

wait to cont

clear
*This section checks for  duplicate NIDs  before entering the
data.

do while check
    do getnid.prg
    use pending index nidpend
    find &NID
    if .not. FOUND()
       store .f. to check
       close all
           clear
       else
           store .t. to check
           text

               The  NID   you  have   tried  to  enter  is  a
duplicate.
               Please check the number and try again.
```

```
              endtext
              wait to cont
              clear
              close all
         endif
    enddo

    use pending index nidpend
    append blank
    replace NID_NO with NID
    set format to pending
    edit for nid_no = NID
    close all

    use property index nid_prop
    append blank
    replace NID_NO with NID
    set format to property
    edit for nid_no = NID
    close all

    use purchase index nid_purc
    append blank
    replace NID_NO with NID
    set format to purchase
    edit for nid_no = NID
    close all


    clear
    return

    *Eof  NT_PEN_R.PRG
```

```
*PENDING.FMT
*Author:      W. T. Key
*Date:        17 July 1987
*Purpose:     Calls for a "P" to be put in the Pend_stat
*             field of PENDING.DBF.
*I/O:         PENDING.DBF
*Called By:   NT_PEN_R.PRG
*Calls:       None


@  6, 16  SAY "Pending Status"
@  6, 33  GET  PENDING->PEND_STAT  FUNCTION "a"  PICTURE "x"
@  8, 15  SAY
"*****************************************************"
@  9, 15  SAY "* Please enter a 'P' in the Pending Status
Field.*"
@ 10, 15  SAY "* The next screen will appear automatically.
  *"
@ 11, 15  SAY
"*****************************************************"

*Eof  PENDING.FMT
```

```
*PROPERTY.FMT
*Author:     W. T. Key
*Date:       3 August 1987
*Purpose:    Allows user to fill those PROPERTY.DBF data
*            fields known before equipment has arrived.
*I/O:        PROPERTY.DBF
*Called By:  NT_PEN_R.PRG
*Calls:      None


● 5,  9  SAY "Mfg Name"
● 5, 28  GET  PROPERTY->MFG_NAME
● 7,  9  SAY "Noun Name"
● 7, 28  GET  PROPERTY->NOUN_NAME
● 9,  9  SAY "Mfg Model Number"
● 9, 28  GET  PROPERTY->MFG_MOD_NO
● 11,  9  SAY "Property Code"
● 11, 28  GET  PROPERTY->PROP_CODE
● 13,  9  SAY "Department Code"
● 13, 28  GET  PROPERTY->DEPT_CODE
● 15,   0   SAY  "** Press  <CTRL> <END> to save the data you
just entered. **"

Eof  PROPERTY.FMT
```

```
*PURCHASE.FMT
*Author:      W. T. Key
*Date:        3 August 1987
*Purpose:     Allows the user to fill in certain PURCHASE.DBF
*             data fields which are known prior to equipment
*             arrival.
*I/O:         PURCHASE.DBF
*Called By:   NT_PEN_R.PRG
*Calls:       None


@  6, 11  SAY "Consignor"
@  6, 26  GET  PURCHASE->CONSIGNOR
@  8, 11  SAY "Date Entered"
@  8, 35  GET  PURCHASE->DATE_RECV
@ 10, 11  SAY "Purchase Order Number"
@ 10, 35  GET  PURCHASE->PO_NO
@ 12, 11  SAY "Requisition Number"
@ 12, 35  GET  PURCHASE->REQN_NO
@ 14, 11  SAY "** For the Date Entered, use today's date. **"
@ 16,  8  SAY "** Press <CTRL> <END> to save the data you
just entered. **"

*Eof  PURCHASE.FMT
```

```
*GETNID.PRG
*Author:        W. T. Key
*Date:          3 August 1987
*Purpose:       Solicits NID number from the user shows the
*               user a template then checks for proper length
*               and alphabetic characters.  It continues to
*               loop until a proper NID number is input.
*I/O:           None
*Called By:     (practically all)
*Calls:         None


clear
set talk off
public NID

store "62271       " to NID
store .t. to check

do while check

@12,05 say [Enter the NID number followed by <RTN>:]
@12,45 get NID picture '99999-999999'
read
    if len(rtrim(NID)) < 12
            store .t. to check
      @14,10 say [The NID number you input was too short;]
      @15,10 say [please check and try again.]
      @16,10 say []
            wait to continue
            clear
        else
            store .f. to check
            clear
    endif
enddo


*Eof GETNID.PRG
```

```
*EQ_RECPT.PRG
*Author:      W. T. Key
*Date:        17 July 1987
*Purpose:     Once an item of Plant or Minor property is
*             received the pending file is checked for
*             accuracy, the date received and mfg's serial
*             number are added.  The result of this process is
*             the WS 1342+ and is now ready to be sent to the
*             destination department.
*I/O:         PENDING.DBF
*Called By:   ACQUISIT.PRG (PMPS.PRG)
*Calls:       GTWS1342.PRG

clear
set talk off
public prop_type
store .t. to check
store .t. to check_var
store " " to prop_type

text
                ***************************************

                At this time you will be requested to
                compare the information in the pending
                file to the information on the piece
                of equipment which actually arrived.

                You may make necessary changes in the
                highlighted fields.

                **************************************

endtext
wait to cont
clear

do while check
    do getnid.prg
    use pending index nidpend
    find &NID
    if .not. FOUND()
        store .t. to check

        text
                ***************************************

                 There is nothing in the pending file
                 with the NID number you just entered.
                 Please check it and try again.

                ***************************************
```

```
            endtext

            wait to cont
            clear
        else
            if upper(pend_stat) = "R"
                store .t. to check
                text

                       ***************************************

                        This item has already been recieved.
                        Please enter a different NID Number.

                       ***************************************
                endtext

                wait to cont
                clear
            else
                store .f. to check
            endif
        endif
    enddo

    do while check_var
    accept "Enter P for Plant or M for Minor Property and <RTN>:
    " to prop_type
        if upper(prop_type) = "P"
            store .f. to check_var
        else
            if upper(prop_type) = "M"
                store .f. to check_var
            else
                store .t. to check_var
                text
               *****************************************

                Only P or M are acceptable characters.
                Please try again.

               *****************************************
               endtext
                wait to cont
                clear
            endif
        endif
    enddo

    do gtws1342.prg

    clear
```

```
return

*Eof  EQ_RECPT.PRG
```

```
*GTWS1342.PRG
*Author:      W. T. Key
*Date:        30 June 1987 (Mod 9/1/87 BAW)
*Purpose:     This module allows MCD to fill in additional
*             data about a piece of equipment once it has
*             arrived.  It calls three dbfs and their
*             attendant screens for ease of data entry.
*I/O:         PROPERTY.DBF, PENDING.DBF, PURCHASE.DBF,
*             PROP_1.FMT,PEND_1.FMT, PURCH_1.FMT.
*Called By:   EQ_RECPT.PRG (ACQUISIT.PRG) ((PMPS.PRG))
*Calls:       None

clear
set talk off

use pending index nidpend
find &NID
replace pend_stat with "R"
set format to pend_1
edit for nid_no = NID
clear

use property index nid_prop
find &NID
set format to prop_1
edit for nid_no = NID
clear

use purchase index nid_purc
find &NID
set format to purch_1
edit for nid_no = NID
clear

close all
return

*Eof  GTWS1342.PRG
```

```
*PEND_1.FMT
*Author:      W. T. Key
*Date:        3 August 1987
*Purpose:     Allows user to change the value in pend_stat
*             from 'P' to 'R'.
*I/O:         PENDING.DBF
*Called By:   GTWS1342.PRG
*Calls:       None


@  6, 14  SAY "NID Number"
@  6, 35  GET  PENDING->NID_NO
@  8, 14  SAY "Pending Status"
@  8, 39  GET  PENDING->PEND_STAT
@ 10, 12  SAY "*************************************"
@ 11, 12  SAY "* Be sure the Pending Status is R.*"
@ 12, 12  SAY "*************************************"
@ 15,  3  SAY "** If everything looks correct, press **"
@ 16,  5  SAY "** <CTRL> <END> to save the data.    **"

Eof  PEND_1.FMT
```

```
*PROP_1.FMT
*Author:        W. T. Key
*Date:          5 August 1987
*Purpose:       Allows user to fill in data fields in the
*               PROPERTY.DBF which cannot normally be filled
*               in until the equipment has arrived.
*I/O:           PROPERTY.DBF
*Called By:     GTWS1342.PRG
*Calls:         None


@  1, 13  SAY "NID Number"
@  1, 32  GET  PROPERTY->NID_NO
@  3, 13  SAY "Mfg Name"
@  3, 32  GET  PROPERTY->MFG_NAME
@  5, 13  SAY "Mfg Serial Number"
@  5, 32  GET  PROPERTY->MFG_SER_NO
@  7, 13  SAY "Noun Name"
@  7, 32  GET  PROPERTY->NOUN_NAME
@  9, 13  SAY "Mfg Model Number"
@  9, 32  GET  PROPERTY->MFG_MOD_NO
@ 11, 13  SAY "Property Code"
@ 11, 32  GET  PROPERTY->PROP_CODE
@ 13, 13  SAY "Department Code"
@ 13, 32  GET  PROPERTY->DEPT_CODE
@ 15,  0  SAY "** Add or change data as needed. **"
@ 17,  0  SAY "** Press <Ctrl> <End> to save the data. **"

Eof  PROP_1.FMT
```

```
*PURCH_1.FMT
*Author:       W. T. Key
*Date:         5 August 1987
*Purpose:      Allows user to fill in data fields in the
*              PURCHASE.DBF which normally cannot be filled
*              until the equipment has arrived.
*I/O:          PURCHASE.DBF
*Called By:    GTWS1342.PRG
*Calls:        None


●  2,  0  SAY "NID Number"
●  2, 23  GET  PURCHASE->NID_NO
●  4,  0  SAY "Consignor"
●  4, 23  GET  PURCHASE->CONSIGNOR
●  6,  0  SAY "Date Received"
●  6, 23  GET  PURCHASE->DATE_RECV
●  8,  0  SAY "Purchase Order Number"
●  8, 23  GET  PURCHASE->PO_NO
● 10,  0  SAY "Requisition Number"
● 10, 23  GET  PURCHASE->REQN_NO
● 12,   0   SAY "** Be sure to change the Date Received field
to the actual Date Received.**"
● 14,  0  SAY "** Add or change other data as needed. **"
● 15,  0  SAY "** Press <Ctrl> <End> to save the data **"

Eof  PURCH_1.FMT
```

```
*MAKE1342.PRG
*Author:      W. T. Key
*Date:        20 July 1987
*Purpose:     Once a piece of equipment has been sent to the
*             destination department and the WS1342 returned,
*             there is enough information to fill out a proper
*             DD1342.  This module is where that function is
*             done.
*I/O:         PLANT.DBF, INV_INFO.DBF, PLANT.FMT, INV_INFO.FMT
*Called By:   ACQUISIT.PRG (PMPS.PRG)
*Calls:       STOCK.PRG

clear
set talk off
text

*****************************************************

 You are now going to enter the rest of the data
 needed for a complete DD1342.  There will be five
 screens with fields to fill in.  Remember to press
 <CTRL><END> to save the information entered.

*****************************************************

endtext

wait to continue

do getnid

use plant index nidplant
append blank
replace nid_no with NID
set format to plant
edit for nid_no = NID
clear

use inv_info index nid_inv
append blank
replace nid_no with NID
set format to inv_info
edit for nid_no = NID
clear
close all

do stock.prg

clear
return

*Eof  MAKE1342.PRG
```

```
*PLANT.FMT
*Author:      W. T. Key
*Date:        3 August 1987
*Purpose:     Provides screen for user to fill the PLANT.DBF
*             data fields known after WS1342 has returned from
*             destination department.
*I/O:         PLANT.DBF
*Called By:   MAKE1342.PRG
*Calls:       None


● 5, 18   SAY "Commodity Code"
● 5, 32   GET  PLANT->COMOD_CODE
● 7,  7   SAY "Length"
● 7, 17   GET  PLANT->LENGTH
● 7, 44   SAY "Mfg Code"
● 7, 54   GET  PLANT->MFG_CODE
● 9,  7   SAY "Non Avail Number"
● 9, 25   GET  PLANT->NON_AVL_NO
● 9, 44   SAY "Power Code"
● 9, 56   GET  PLANT->PWR_CODE
● 11,  7  SAY "Weight"
● 11, 17  GET  PLANT->WEIGHT
● 11, 33  SAY "Width"
● 11, 40  GET  PLANT->WIDTH
● 11, 55  SAY "Height"
● 11, 62  GET  PLANT->HEIGHT
● 15,  4  SAY  "** Press  <CTRL> <END>  to save  the data you
just entered. **"

Eof  PLANT.FMT
```

```
*INV_INFO.FMT
*Author:        W. T. Key
*Date:          3 August 1987
*Purpose:       Provides data entry screen for the
*               INV_INFO.DBF; called after WS1342 has returned
*               from the destination department.
*I/O:           INV_INFO.DBF
*Called By:     MAKE1342.PRG
*Calls:         None


@  6, 18  SAY "Inventory Date"
@  6, 36  GET  INV_INFO->INV_DATE
@  8, 18  SAY "Julian Date"
@  8, 36  GET  INV_INFO->JULIAN
@ 10, 18  SAY "Location"
@ 10, 36  GET  INV_INFO->LOCATE
@ 14, 2  SAY "** Press <CTRL> <END> to save the data you just
entered. **"

*Eof  INV_INFO.FMT
```

```
*STOCK.PRG
*Author:       W. T. Key
*Date:         13 July 1987 (Mod 9/1/87 BAW)
*Purpose:      This program is called after equipment  has gone
*              to the destination department and the WS1342 has
*              come back with additional data added by the
*              department.  It also checks the stock number for
*              duplication to preclude duplicating information
*              IN the physical description field of
*              ITEM_DES.DBF.  If the stock no. is new it
*              updates both MFG_INFO and ITEM_DES, if it's a
*              dupe it checks to be sure that the duplication
*              is intentional and just updates the MFG_INFO
*              database.
*I/O:          MFG_INFO.DBF, ITEM_DES.DBF; MFG_INFO.FMT,
*              ITEM_DES.FMT
*Called By:    MAKE1342.PRG (ACQUISIT.PRG) ((PMPS.PRG))
*Calls:        None

set talk off

store .f. to not_found
store 'y' to check
store "     " to stk_no

do while .not. not_found
    clear
    accept  "Enter  the  Stock  Number of the equipment: " to
stk_no
    use item_des
    locate for stock_no = stk_no
    if eof()
        store .t. to not_found
        close all
        use mfg_info index nid_mfg
        append blank
        replace nid_no with NID
        replace stock_no with stk_no
        set format to mfg_info
        edit for nid_no = NID
        clear

        use item_des
        append blank
        replace stock_no with stk_no
        set format to item_des
        edit for nid_no = NID
        clear
        close all
    else
        text
          Are you sure the Stock Number &stk_no is correct?
        endtext
        accept "Yes or No? " to check
        clear
        if upper(check) = 'Y'
            store .t. to not_found
            close all databases
            use mfg_info index nid_mfg
            append blank
            replace nid_no with NID
            replace stock_no with stk_no
            set format to mfg_info
            edit for stock_no = stk_no
            clear
            close all
        else
            close all
            store .f. to not_found
            text
This stock number &stk_no is already in the database.
Please check the number and try again.
            endtext
          wait to cont
        endif
    endif
enddo

clear
return


*Eof STOCK.PRG
```

```
*MFG_INFO.FMT
*Author:      W. T. Key
*Date:        3 August 1987
*Purpose:     Presents screens for data input into the
*             MFG_INFO.DBF.  Calls for data which is not
*             normally available until the WS1342 is back
*             from the destination department.
*I/O:         MFG_INFO.DBF
*Called By:   STOCK.PRG
*Calls:       None


@  5, 17  SAY "Mfg Year"
@  5, 32  GET  MFG_INFO->MFG_YEAR
@  7, 17  SAY "Type Code"
@  7, 32  GET  MFG_INFO->TYPE_CODE
@  9, 17  SAY "Cost"
@  9, 32  GET  MFG_INFO->COST
@ 13,  7  SAY  "** Press  <CTRL> <END>  to save  the data you
just entered. **"

Eof  MFG_INFO.FMT
```

```
*ITEM_DES.FMT
*Author:       W. T. Key
*Date:         5 August 1987
*Purpose:      Presents screen for user to fill in the
*              physical description of the item--by stock
*              number--ie several NID numbers may have the
*              stock_no and thus only one phys_descr is nesc.
*I/O:          ITEM_DES.DBF
*Called By:    STOCK.PRG
*Calls:        None


● 2,30  SAY "Physical Description"
● 4,01  GET ITEM_DES->PHYS_DESC
●  10,  01    SAY  [Enter  the  equipment  description in the
highlighted]
● 11, 01  SAY [area.    Use  the  left/right  arrows  to move
around]
● 12, 01  SAY [in the physical description field.]
●  14,  01    SAY  [Press  <CTRL><END>  to  save the data you
entered.]


Eof  ITEM_DES.FMT
```

```
*PRNT1342.PRG
*Author:     W. T. Key
*Date:       14 July 1987 (Mod 9/15/87 BAW)
*Purpose:    Prints a form DD1342 with inputs from
*            all necessary dbfs with the same NID no.
*I/O:        PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF,
*            INV_INFO.DBF,
*            PURCHASE.DBF; PRNT1342.FMT
*Called By:  ACQUISIT.PRG (PMPS.PRG)
*Calls:      None


clear
set talk off
erase temp_1.dbf
erase temp_2.dbf
erase temp_3.dbf
erase temp_A.dbf

do getnid.prg

use property index nid_prop
find &NID
if .not. FOUND() then
   @ 10,10 say [Record does not exist!]
   ?
   wait
   return
endif
go top
select 2
use plant
select 1
join with plant to temp_1 for nid_no = plant -> nid_no
close all


use temp_1
select 2
use mfg_info
select 1
join with mfg_info to temp_2 for nid_no = mfg_info -> nid_no
close all


use temp_2
select 2
use inv_info
select 1
join with inv_info to temp_3 for nid_no = inv_info -> nid_no
close all
```

```
use temp_3
select 2
use purchase
select 1
join  with  purchase  to  temp_A  for nid_no=purchase->nid_no
fields
nid_no,julian,stock_no,cost,dept_code,mfg_year,mfg_name,-
mfg_code-
,mfg_mod_no,comod_code,type_code,non_avl_no,pwr_code,length,-
width,height,weight,noun_name,po_no,mfg_ser_no,locate

close all
use temp_A
locate for nid_no = NID
clear
set device to print

do prnt1342.fmt
eject
set device to screen
close all databases

erase temp_1.dbf
erase temp_2.dbf
erase temp_3.dbf
erase temp_A.dbf

wait to continue
clear

return


*Eof  PRNT1342.PRG
```

```
*VIEW1342.PRG
*Author:      W. T. Key
*Date:        20 July 1987 (Mod 9/1/87 BAW)
*Purpose:     Draws a filled-in DD1342 on the screen for the
*             user to browse through and make changes as
*             necessary.  File selected via NID number input
*             by user.  Changes made on the screen will be
*             stored in the appropriate dbf.
*I/O:         PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF,
*             INV_INFO.DBF
*             PURCHASE.DBF; VU1342P1.FMT, VU1342P2.FMT
*Called By:   ACQUISIT.PRG (PMPS.PRG)
*Calls:       None


erase temp_1.dbf
erase temp_2.dbf
erase temp_3.dbf
erase temp_B.dbf
clear
set talk off

do getnid

use property index nid_prop
find &NID
if .not. FOUND() then
   @10,10 say [Record does not exist!]
   ?
   wait
   return
endif
go top
select 2
use plant
select 1
join with plant to temp_1 for nid_no = plant -> nid_no
close all

use temp_1
select 2
use mfg_info
select 1
join with mfg_info to temp_2 for nid_no = mfg_info -> nid_no
close all

use temp_2
select 2
use inv_info
select 1
join with inv_info to temp_3 for nid_no = inv_info -> nid_no
```

```
close all

use temp_3
select 2
use purchase
select 1
join with  purchase to temp_B for nid_no = purchase -> nid_no
fields nid_no,;
julian,stock_no,cost,dept_code,mfg_year,mfg_name,mfg_code,-
mfg_mod_no,;
mfg_ser_no,locate,comod_code,type_code,non_avl_no,pwr_code,-
length,;
width,height,weight,noun_name,po_no

erase temp_1.dbf
erase temp_2.dbf
erase temp_3.dbf
close all

use temp_B
go top

locate for nid_no = NID

clear

set format to vu1342p1

    read

    select 2
    use property index nid_prop
    find &NID
    replace dept_code with temp_B -> dept_code
    replace mfg_name with temp_B -> mfg_name
    replace mfg_mod_no with temp_B -> mfg_mod_no
    replace mfg_ser_no with temp_B -> mfg_ser_no

    select 3
    use plant index nidplant
    find &NID
    replace comod_code with temp_B -> comod_code
    replace pwr_code with temp_B -> pwr_code
    replace mfg_code with temp_B -> mfg_code

    select 4
    use mfg_info index nid_mfg
    find &NID
    replace stock_no with temp_B -> stock_no
    replace cost with temp_B -> cost
    replace mfg_year with temp_B -> mfg_year
    replace type_code with temp_B -> type_code
```

```
        select 5
        use inv_info index nid_inv
        find &NID
        replace julian with temp_B -> julian

        select 1
        set format to vu1342p2
        read

        select 2
        replace noun_name with temp_B -> noun_name
        replace dept_code with temp_B -> dept_code

        select 3
        replace length with temp_B -> length
        replace width with temp_B -> width
        replace height with temp_B -> height
        replace weight with temp_B -> weight
        replace non_avl_no with temp_B -> non_avl_no

        select 4
        use purchase index nid_purc
        find &NID
        replace po_no with temp_B -> po_no

        select 5
        replace locate with temp_B -> locate

     close all

    erase temp_B.dbf

    clear
    return


    *Eof  VIEW1342.PRG
```

```
*VU1342P1.FMT
*Author:       W. T. Key
*Date:         3 August 1987
*Purpose:      Builds a screen that looks like the top half of
*              a DD1342 and fills it with data from below
*              noted DBFs.  The user may then browse and
*              change as desired with changes being stored in
*              appropriate DBFs.
*I/O:          PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF,
*              INV_INFO.DBF,
*              PURCHASE.DBF
*Called By:    VIEW1342.PRG
*Calls:        None


    @1,1 say [DOD Property]
    @1,16 say [1._X_Active]
    @1,28 say [_X_Initial]
    @1,39 say [2.Julian]
    @1,49 say [3.ID/Govt Tag#]
    @1,65 say [Form Approved]
    @2,4 say  [Record]
    @2,20 say [__Idle]
    @2,30 say [__Change]
    @2,41 say [Date]
    @2,65 say [OMB No.22-R0209]
    @3,41 get julian
    @3,51 get nid_no
    @4,0 to 4,79
    @5,25 say [Section 1--INVENTORY RECORD]
    @5,55 say [Dept:]
    @5,61 get dept_code
    @6,0 to 6,79
    @8,1 say  [4.Commodity Code]
    @8,19 say [5.Stock]
    @8,29 say [6.Acqn Cost]
    @8,42 say [7.Type]
    @8,50 say [8.Yr of]
    @8,60 say [9.Pwr]
    @9,21 say [Number]
    @9,44 say [Code]
    @9,52 say [Mfg]
    @9,62 say [Code]
    @10,2 get comod_code
    @10,21 get stock_no
    @10,30 get cost
    @10,45 get type_code
    @10,52 get mfg_year
    @10,63 get pwr_code
    @12,00 to 12,79
    @13,1  say [10.Status]
    @13,12 say [11.SVC]
    @13,21 say [12.Command]
    @13,33 say [13.ADM Office]
    @14,4  say [Code]
    @14,15 say [Code]
    @14,24 say [Code]
    @14,36 say [Code]
    @15,4  say [1A]
    @15,15 say [1]
    @15,23 say [N00011]
    @15,36 say [N00228]
    @16,00 to 16,79
    @17,1  say [14.Name of]
    @17,25 say [15.Mfr's]
    @17,35 say [16.Manufacturer's]
    @17,58 say [17.Manufacturer's]
    @18,4  say [Manufacturer]
    @18,28 say [Code]
    @18,38 say [Model No.]
    @18,61 say [Serial No.]
    @19,3  get mfg_name
    @19,28 get mfg_code
    @19,38 get mfg_mod_no
    @19,60 get mfg_ser_no

*Eof  VU1342P1.FMT
```

```
*VU1342P2.FMT
*Author:       W. T. Key
*Date:         1 August 1987
*Purpose:      Builds a acreen that looks like the bottom half
*              of a DD1342 and fills it with data from the
*              below noted DBFs.  The user may then browse and
*              change as desired with changes stored in the
*              appropriate DBF.
*I/O:          PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF,
*              INV_INFO.DBF,PURCHASE.DBF
*Called By:    VIEW1342.PRG
*Calls:        None


    @3,1 say [18.Length]
    @3,11 say [19.Width]
    @3,20 say [20.Height]
    @3,30 say [21.Weight]
    @3,40 say [22.Certificate of]
    @3,58 say [25.Contract No.]
    @4,43 say [Non-Avail No.]
    @5,3  get length
    @5,12 get width
    @5,22 get height
    @5,32 get weight
    @5,42 get non_avl_no
    @5,60 get po_no
    @6,0 to 6,79
    @7,1 say [26.Description and Capacity]
    @9,4 get noun_name
    @11,4 say [phys_desc will go here]
    @13,0 to 13,79
    @14,1 say [28.Present Location]
    @14,32 say [Equip. Location]
    @14,57 say [29.Possessor]
    @15,4 say [Naval Postgraduate School]
    @15,32 get locate
    @15,60 say [Code]
    @16,4 say [No Street Address]
    @16,60 say [62271(7685435)]
    @17,4 say [Monterey, Ca.]


*Eof  VU1342P2.PRG
```

```
*INVMENU.PRG
*Author:      W. T. Key
*Date:        26 July 1987
*Purpose:     This module provides access to the two inventory
*             sub-modules; it is a simple menu which allows
*             the user to print or browse the inventory files
*             (in DD1342(-) format) or return to the
*             Acquisitions
*             Menu.
*I/O:         None
*Called By:   ACQUISIT.PRG (PMPS.PRG)
*Calls:       VIEWINV.PRG, PRNTINV.PRG

clear
store "    " to choice

do while .t.
    @8,30 say [Conduct Inventory Menu]
    @10,23 say [Task                                Task Code]
    @12,23 say [Browse/Edit Inventory List              1]
    @14,23 say [Print Inventory List                    2]
    @16,23 say [Return to Previous Menu                 3]
    ?
    wait "Enter Task Code: " to choice

    do case
        case choice = '1'
            do viewinv.prg
        case choice = '2'
            do prntinv.prg
        case choice = '3'
            clear
            return
        otherwise
            loop
    endcase
enddo

return

*Eof  INVMENU.PRG
```

```
*VIEWINV.PRG
*Author:      W. T. Key
*Date:        29 July 1987
*Purpose:     This program allows the user to browse thru
*             screens made up of DD1342(-) fields.  It calls
*             up, then creates a screen for each piece of
*             equipment which meets the prompted filters of
*             inventory cutoff date and department of
*             interest.
*             Changes made on the screen will be stored in
*             the appropriate dbf.
*I/O:         PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF,
*             INV_INFO.DBF; INVRPT.FMT
*Called By:   INVMENU.PRG (ACQUISIT.PRG) ((PMPS.PRG))
*Calls:       None


erase temp_1.dbf
erase temp_2.dbf
erase invrpt.dbf

clear
set talk off

accept "Enter inventory cutoff date (mm/dd/yy) and <RTN>:" to
input_1
accept "Enter department code  (4-digit)  and  <RTN>:    " to
input_2

use property
select 2
use plant
select 1
join with plant to temp_1 for nid_no = plant -> nid_no
close all

use temp_1
select 2
use mfg_info
select 1
join with mfg_info to Temp_2 for nid_no = mfg_info -> nid_no
close all

use temp_2
select 2
use inv_info
select 1
join with inv_info to invrpt for nid_no = inv_info -> nid_no
fields  nid_no,  inv_date,  noun_name,  locate,  mfg_ser_no,
mfg_code,
comod_code, stock_no, cost, dept_code, mfg_name
```

```
close all

use invrpt
go top

do while .not. eof()
    if UPPER(dept_code) = UPPER(input_2) .and. DTOC(inv_date)
< input_1

        set format to invrpt

        read

        select 2
        use property
        locate for property->nid_no = invrpt->nid_no
        replace mfg_name with invrpt->mfg_name
        replace noun_name with invrpt->noun_name
        replace dept_code with invrpt->dept_code
        replace mfg_ser_no with invrpt->mfg_ser_no

        select 3
        use plant
        locate for plant->nid_no = invrpt->nid_no
        replace mfg_code with invrpt->mfg_code
        replace comod_code with invrpt->comod_code

        select 4
        use mfg_info
        locate for mfg_info->nid_no = invrpt->nid_no
        replace stock_no with invrpt->stock_no
        replace cost with invrpt->cost

        select 5
        use inv_info
        locate for inv_info->nid_no = invrpt->nid_no
        replace inv_date with invrpt->inv_date
        replace locate with invrpt->locate

       select 1

        skip

    else

        skip

    endif
enddo

erase temp_1.dbf
erase temp_2.dbf
erase invrpt.dbf
close all
clear
return

*Eof  VIEWINV.PRG
```

```
*PRNTINV.PRG
*Author:      W. T. Key
*Date:        1 August 1987
*Purpose:     This module allows the user to print an
*             equipment inventory which displays the
*             DD1342(-) fields.  The user is prompted
*             for an inventory cutoff date and department
*             of interest.
*I/O:         PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF,
*             INV_INFO.DBF; INVRPT_P.FMT
*Called By:   INVMENU.PRG (ACQUISIT.PRG) ((PMPS.PRG))
*Calls:       None


erase temp_1.dbf
erase temp_2.dbf
erase invrpt.dbf

clear
set talk off

accept "Enter inventory cutoff date: (mm/dd/yy)  " to input_1
accept "Enter department code (4-digit):  " to input_2

use property
select 2
use plant
select 1
join with plant to temp_1 for nid_no = plant -> nid_no
close all

use temp_1
select 2
use mfg_info
select 1
join with mfg_info to Temp_2 for nid_no = mfg_info -> nid_no
close all

use temp_2
select 2
use inv_info
select 1
join with inv_info to invrpt for nid_no = inv_info -> nid_no
fields  nid_no,  inv_date,  noun_name,   locate,  mfg_ser_no,
mfg_code,
comod_code, stock_no, cost, dept_code, mfg_name
close all

use invrpt
go top
@ 0,0
```

```
set device to print

do while .not. eof()
    if UPPER(dept_code) = UPPER(input_2) .and. DTOC(inv_date)
< input_1

        @1,26 say [MCD Inventory Worksheet]
        @3,0 say [NID No.]
        @3,22 say nid_no
        @5,0 say [Last Inventory]
        @5,22 say inv_date
        @7,0 say [Department Code]
        @7,22 say dept_code
        @9,0 say [Equipment Name]
        @9,22 say noun_name
        @11,0 say [Location of Equipment]
        @11,22 say locate
        @13,0 say [Mfg's Serial No.]
        @13,22 say mfg_ser_no
        @15,0 say [Commodity Code]
        @15,22 say comod_code
        @16,0 say [Stock No.]
        @16, 22 say stock_no
        @17,0 say [Acquisition Cost]
        @17,22 say cost
        @18,0 say [Mfg Name]
        @18,22 say mfg_name

skip

        loop
    else

        skip
        loop
    endif

enddo
eject
set device to screen

close all
erase temp_1.dbf
erase temp_2.dbf
erase invrpt.dbf
clear
return


*Eof  PRNTINV.PRG
```

```
*PENDMENU.PRG
*Author:      W. T. Key
*Date:        22 July 1987
*Purpose:     This program allows the user access to the
*             pending sub-modules.  It is a simple menu which
*             gives the user the choice of browsing or
*             printing pending file or return to the
*             Acquisitions Menu.
*I/O:         None
*Called By:   ACQUISIT.PRG (PMPS.PRG)
*Calls:       VIEWPEND.PRG, PRNTPEND.PRG


clear

store "    " to choice

do while .t.

●  8,30 say [PENDING Files MENU]
● 10,23 say [Task                                  Task Code]
● 12,23 say [Browse/Edit Pending Files                 1]
● 14,23 say [Print Pending Files                       2]
● 16,23 say [Return to Previous Menu                   3]

?

wait "Enter Task Code:   " to choice

do case
    case choice = "1"
        do viewpend.prg

    case choice = "2"
        do prntpend.prg

    case choice = "3"
        clear
        exit

    otherwise
        loop

    endcase
enddo


*Eof  PENDMENU.PRG
```

```
*VIEWPEND.PRG
*Author:      W. T. Key
*Date:        25 July 1987
*Purpose:     Displays selected fields for all items in the
*             "pending file" ie those pieces of equipment
*             carrying a 'P' (or an 'R') in the pend_stat
*             field.  The user is prompted for a 'P'
*             or an 'R'.  This module could sort on anything
*             in the pend_stat field. Changes made on the
*             screen will be stored in the appropriate dbf.
*I/O:         PROPERTY.DBF, PURCHASE.DBF, PENDING.DBF,
*             PENDRPT1.FMT
*Called By:   PENDMENU.PRG (ACQUISIT.PRG) ((PMPS.PRG))
*Calls:       None


clear
set talk off

erase pendrpt1.dbf
erase temp_1.dbf

accept "Enter equipment status (P or  R)  and  <RTN>:    " to
input

use property
select 2
use purchase
select 1
join with purchase to temp_1 for nid_no = purchase -> nid_no
close all

use temp_1
select 2
use pending
select 1
join with  pending to pendrpt1 for nid_no = pending -> nid_no
fields nid_no, reqn_no, dept_code, noun_name, pend_stat,
date_recv, po_no

close all
use pendrpt1

go top

do while .not. eof()
    if pendrpt1->pend_stat = input

        set format to pendrpt1

        read
        select 2
        use property
        locate for property->nid_no = pendrpt1->nid_no
        replace property->noun_name with pendrpt1->noun_name
        replace property->dept_code with pendrpt1->dept_code

        select 3
        use purchase
        locate for purchase->nid_no = pendrpt1->nid_no
        replace purchase->date_recv with pendrpt1->date_recv
        replace purchase->reqn_no with pendrpt1->reqn_no
        replace purchase->po_no with pendrpt1->po_no

        select 4
        use pending
        locate for pending->nid_no = pendrpt1->nid_no
        replace pending->pend_stat with pendrpt1->pend_stat

        select 1

        skip

    else
        skip

    endif

enddo

erase temp_1.dbf

erase pendrpt1.dbf

close all

clear

return

*Eof  VIEWPEND.PRG
```

```
*PRNTPEND.PRG
*Author:      W. T. Key
*Date:        22 July 1987
*Purpose:     Prints all items which have 'P' or 'R' in the
*             pend_stat field.  User is prompted for 'P/R' as
*             desired.
*I/O:         PROPERTY.DBF, PURCHASE.DBF, PENDING.DBF;
*             no *.FMTs)
*Called By:   PENDMENU.PRG (ACQUISIT.PRG) ((PMPS.PRG))
*Calls:       None


clear

set talk off

erase pendrpt1.dbf
erase temp_1.dbf


accept "Enter equipment status (P or  R)  and  <RTN>:     " to
input

use property
select 2
use purchase
select 1
join with purchase to temp_1 for nid_no = purchase -> nid_no
close all

use temp_1
select 2
use pending
select 1
join with  pending to pendrpt1 for nid_no = pending -> nid_no
fields nid_no, reqn_no, dept_code, noun_name, pend_stat,
date_recv, po_no

use pendrpt1

go top

@ 0,0

set device to print

do while .not. eof()

  if pendrpt1-> pend_stat = input

    @ 1,30 say [MCD Pending Report]
    @ 4,0 say "NID_NO"
    @ 4,23 say pendrptl -> nid_no
    @ 6,0 say "Requisition Number"
    @ 6,23 say pendrptl -> REQN_NO
    @ 8,0 say "Department Code"
    @ 8,23 say pendrptl -> DEPT_CODE
    @ 10,0 say "Item Name"
    @ 10,23 say pendrptl -> NOUN_NAME
    @ 12,0 say "Pending Status"
    @ 12,23 say pendrptl -> PEND_STAT
    @ 14,0 say "Date Received"
    @ 14,23 say pendrptl -> DATE_RECV
    @ 16,0 say "Purchase Order No."
    @ 16,23 say pendrptl -> PO_NO
    skip
    loop

  else
    skip
    loop
 endif

enddo

eject

set device to screen

close all
clear
return


*Eof PENDPRNT.PRG
```

APPENDIX C

DISPOSITION MODULES SOFTWARE CODE

The following contains the software code contained in the modules which support the equipment dispositions process at MCD. Each module begins with a documentaion header. This header contains the name of the author, the date that the code was last modified, and a general description of the purpose of the module. Additionally listed are all of the programs, formats, and data base files (.PRG, .FMT, .DBF respectively) that the module uses, calls, or is called by.

```
*PROC_TRN.PRG
*Author:        B. A. Whitehouse
*Date:          1 July 1987
*Purpose:       This module calls two modules to be executed
*               and return control to the main control
*               module, PMPS.
*I/O Files:     None
*Called by:     PMPS.PRG
*Calls:         INPUT_TR.PRG, BLD_DPEN.PRG

clear
set talk off

do input_tr
do bld_dpen

close all
return

*eof PROC_TRN.PRG
```

```
*INPUT_TR.PRG
*Author:        B. A. Whitehouse
*Date:          2 July 1987
*Purpose:       If the user has a PROPNID_NO then this module
*               will execute INPT_EXC. If not then OTHEREXC
*               will be executed.  This will be done for as
*               many times as the user has transactions to
*               input.
*I/O Files:     None
*Called By:     PROC_TRN.PRG
*Calls:         INPT_EXC.PRG, OTHEREXC.PRG

clear
set talk off
store "Y" to answer

do while answer = "Y" .or. answer = "y"
   clear
   ?
   ?
   ?
   wait  "      Do you have a NID Number? (Y/N)"  to answer

   if answer = "y" .or. answer = "Y" then
      do inpt_exc
   else
      clear
      @ 10,5 say [Do you want to process an item without a
NID number?]
      @ 10,58 get answer
      read
      if answer = "Y" .or. answer = "y" then
         do otherexc
      endif
   endif

   clear
   ?
   ?
   ?
   wait "      Do you want to process another transaction?
(Y/N)"
to answer

enddo

clear
close all
return

eof INPUT_TR.PRG
```

```
*INPT_EXC.PRG
*Author:        B. A. Whitehouse
*Date:          14 August 1987
*Purpose:       This module prompts the user for information
*               on an item having been identified as either
*               plant or minor property, having a NID
*               number, and being excess equipment.  The data
*               recieved will be used to build a pending
*               disposition record.
*I/O Files:     PROPERTY.DBF, PEND-D.DBF, PENDING.DBF,
*               NID_PROP.NDX, NID_DPEN.NDX, RPT_DPEN.NDX,
*               NIDPEND.NDX
*Called by:     INPUT_TR.PRG
*Calls:         GETDATA.PRG

clear

*memory variables
again     = space(1)
prop_cd   = space(1)
cond_cd   = space(1)
poc_name  = space(20)
poc_phone = space(4)
qty_exc   = 0
*memory variables
store .t. to again
clear

do while .t.

   do getnid
   clear
   @ 7,5 say "Enter condition code:" get cond_cd
   @ 9,5 say "Enter point of contact:" get poc_name
   @ 11,5 say "Enter POC phone:" get poc_phone
   @ 13,5 say "Enter quanty of excess:" get qty_exc
   @ 15,5 say "Enter property code:" get prop_cd

   read

   select 1
   use property index nid_prop
   find &NID
   if .not. FOUND() then
      append blank
      replace nid_no with NID
      replace prop_code with prop_cd
      do getdata
   endif

   select 4
   use pend-d index nid_dpen,rpt_dpen
   find &NID

   if FOUND() then
      clear
      @ 3,10 say "Record already exists, check NID number and
try again."
      ?
      ?
      wait
   else
      exit
   endif

enddo

append blank

replace cond_c_e  with cond_cd
replace poc_n     with poc_name
replace poc_p_no  with poc_phone
replace nid_no    with NID
replace qty_e     with qty_exc

clear
@ 10,5 say [Check entries on followng screen, if correct
press <ESC>.]
wait
edit for nid_no = NID

clear
select 3
use pending index nidpend
find &NID

if .not. found() then
   append blank
   replace nid_no with NID
   replace pend_stat with "I"
else
   replace pend_stat with "I"
endif

close all
clear
return

*eof INPT_EXC.PRG
```

```
*GETDATA.PRG
*Author:        B. A. Whitehouse
*Date:          14 August 1987
*Purpose:       To load data for equipment that is not
*               currently in the data base.  This data is
*               considered the minimum information necessary
*               to process a particular item for disposition.
*I/O files:     MFG_INFO.DBF, NID_MFG.NDX, PROPERTY.DBF,
*               NID_PROP.NDX
*Called by:     INPT_EXC.PRG
*Calls:         None

clear

*memory variables
store 0.00 to price
store "    " to mfr_yr
store "    " to stock
store "    " to dept
mfr_mod  = space(15)
mfr_name = space(30)
ser_no   = space(20)
n_name   = space(30)
bldg     = space(4)
rm       = space(4)
* memory variables

   ● 1,15 say "Additional Info Is Necessary"
   ● 3,5 say "Enter cost of the item:" get price
   ● 5,5 say "Enter year manufactured, if know:" get mfr_yr
   ● 7,5 say "Enter Stock number:" get stock
   ● 9,5 say "Enter Dept. code:" get dept
   ● 11,5 say "Enter Model Number:" get mfr_mod
   ● 13,5 say "Enter Manufacture's Name:" get mfr_name
   ● 15,5 say "Enter Serial Number:" get ser_no
   ● 17,5 say "Enter Item Name:" get n_name
   ● 19,5 say "Enter Location:"
   ● 19,21 say "BLDG " get bldg
   ● 20,21 say "Room " get rm
   read

   replace dept_code with dept
   replace mfg_mod_no with mfr_mod
   replace mfg_name with mfr_name
   replace mfg_ser_no with ser_no
   replace noun_name with n_name

   select 5
   use mfg_info index nid_mfg
   append blank

   replace nid_no with NID
   replace cost with price
   replace mfg_year with mfr_yr
   replace stock_no with stock

use
select 8
use inv_info index nid_inv
append blank
replace nid_no with NID
replace locate with bldg-rm
use
select 1

clear
return

*eof GETDATA.PRG
```

```
*OTHEREXC.PRG
*Author:        B. A. Whitehouse
*Date:          14 August 1987
*Purpose:       To load the data fields with information
*               provided by the user.  These excess items will
*               be identified by a combination of the date
*               they are entered and their stock number.  This
*               character string will be entered into the
*               NID Number field.
*I/O files:     PROPERTY.DBF, PEND-D.DBF, MFG_INFO.DBF,
*               PENDING.DBF, NID_PROP.NDX, NID_DPEN.NDX,
*               RPT_DPEN.NDX, NIDPEND.NDX
*Called by:     INPUT_TR.PRG
*Calls:         None

set talk off
clear

*Memory variables
   mfr_n     = space(20)
   mfr_ser   = space(20)
   prop_cd   = space(1)
   cond_cd   = space(1)
   poc_name  = space(20)
   poc_phone = space(4)
   qty_exc   = space(3)
   stock     = space(4)
   dept      = space(4)
   mod_no    = space(15)
   mfr_yr    = space(3)
   noun_n    = space(30)
   type_cd   = space(1)
   rm        = space(4)
   bldg      = space(4)
   store 0 to qty
   store 0 to price
   store DTOC(DATE()) to date_ent
* Memory variables

text
    In the following two screens enter as much of the data as
possible.  Since the items are not Plant or Minor Property
or because they are Minor Property which was never assigned
a NID Number, these items will be identifed in this database
by a combination of the date they are entered an their stock
number in the NID Number field.
endtext
?
?
wait
clear
do while .T.
```

```
   @ 5,5 say "Enter condition code:" get cond_cd
   @ 7,5 say "Enter point of contact:" get poc_name
   @ 9,5 say "Enter POC phone:" get poc_phone
   @ 11,5 say "Enter property code:" get prop_cd
   @ 13,5 say "Enter stock number:" get stock
   @ 15,5 say "Enter cost:" get price
   @ 17,5 say "Enter Type Code:" get type_cd

   read
   clear

   @ 5,5 say "Enter Department code:" get dept
   @ 7,5 say "Enter manufacture model number:" get mod_no
   @ 9,5 say "Enter manufacture year:" get mfr_yr
   @ 11,5 say "Enter noun name:" get noun_n
   @ 13,5 say "Enter manufacture's name:" get mfr_n
   @ 15,5 say "Enter serial number:" get mfr_ser
   @ 17,5 say "Enter Item Quantity:" get qty

   read
   clear

   @ 5,5 say "Enter Item Location:"
   @ 5,27 say "BLDG " get bldg
   @ 6,27 say "ROOM " get rm

   read
   clear

   if stock <> "    " .and. bldg <> "    " .and. rm <> "    "
then
      exit
   else
      @ 19,3 say "You must enter the Stock Number and
Location."
   endif
enddo

   select 1
   use property index nid_prop
   select 4
   use pend-d index nid_dpen,rpt_dpen
   select 5
   use mfg_info index nid_mfg
   select 3
   use pending index nidpend
   select 8
   use inv_info index nid_inv
   select 4

   append blank
   replace cond_c_e with cond_cd
   replace poc_n    with poc_name
   replace poc_p_no with poc_phone
   replace nid_no   with date_ent-stock
   replace qty_e    with qty
   select 5
   append blank
   replace mfg_year with mfr_yr
   replace stock_no with stock
   replace cost     with price
   replace nid_no   with date_ent-stock
   replace type_code with type_cd
   select 1
   append blank
   replace dept_code  with dept
   replace mfg_name   with mfr_n
   replace mfg_ser_no with mfr_ser
   replace mfg_mod_no with mod_no
   replace prop_code  with prop_cd
   replace noun_name  with noun_n
   replace nid_no     with date_ent-stock
   select 3
   append blank
   replace nid_no with date_ent-stock
   replace pend_stat with 'I'
   select 8
   append blank
   replace nid_no with date_ent-stock
   replace locate with bldg-rm

close all
clear
return

*eof OTHEREXC.PRG
```

```
*BLD_DPEN.PRG
*Author:        B. A. Whitehouse
*Date:          25 July 1987
*Purpose:       This module first varifies that the number of
*               days in each of the screening periods has not
*               changed.  It then completes each pending
*               disposition record.  That is, those records
*               with an "I" in the PEND_STAT field.  The
*               termination date is initially determined by
*               the type of equipment,Industrial Plant
*               Equipment, ADPE, or no special type.  Other
*               data filled in are PHASE, REPORT_NO,
*               DATE_ENT_D and the PEND_STAT is changed
*               to indicate a completed pending dispositions.
*I/O files:     PEND-D.DBF, PROPERTY.DBF, PENDING.DBF,
*               MFG_INFO.DBF, NID_DPEN.NDX, RPT_DPEN.NDX,
*               NID_PROP.NDX, NIDPEND.NDX, NID_MFG.NDX,
*               TEMPSF.DBF, SF120REP.FRM
*Called By:     PROC_TRN.PRG
*Calls:         CALJUL.UTL, 1342IREP.PRG, SF120REP.PRG,
*               HTLSTREP.PRG

clear

*memory variables
r_scr_p     =  21
ie_scr_p    =  90
adpe_scr_p = 201
uic         = 'N62271'
ans         = ' '

@ 5,7 say "Enter current IE screening period:" get ie_scr_p
@ 7,7 say "Enter current Reutilization screening period:" get
r_scr_p
@ 9,7 say "Enter current ADPE screening period:" get
adpe_scr_p

read
clear

select 4
use pend-d index nid_dpen,rpt_dpen
select 1
use property index nid_prop
select 4
set relation to nid_no into property
select 3
use pending index nidpend
select 1
set relation to nid_no into pending
select 5
use mfg_info index nid_mfg
```

```
select 3
set relation to nid_no into mfg_info

do caljul.utl
select 4
go top

do while .not. EOF()
   if SUBSTR(nid_no,3,1) <> "/" .and. pending->pend_stat =
"I" then
      replace date_ent_d with julian
      replace rpt_no with uic-"-"-LTRIM(STR(julian))
      replace phase with "R"
      select 5
      if mfg_info->type_code = "1" then
         select 4
         Mdate_t = julian + ie_scr_p
         do chkjul with Mdate_t
         replace date_term with Mdate_t

         select 8
         use temp_nid
         append blank
         store pend-d->nid_no to nid_no
         use

         select 3
         replace pend_stat with "D"
         select 4
      else
         if VAL(stock_no) >= 7000 .and. VAL(stock_no) < 8000
then
            select 4
            Mdate_t = julian + adpe_scr_p
            do chkjul with Mdate_t
            replace date_term with Mdate_t
            do sf120rep
            select 4
            replace pending->pend_stat with "D"
         else
            select 4
            Mdate_t = julian + r_scr_p
            do chkjul with Mdate_t
            replace date_term with Mdate_t
            select 3
            replace pend_stat with "D"
            select 4
         endif
      endif
    else
      if pending->pend_stat = "I"
         replace date_ent_d with julian
```

```
          replace rpt_no with uic-"-"-LTRIM(STR(julian))
          replace phase with "R"
          replace pending->pend_stat with "D"
       endif
    endif
skip
enddo

close all
clear
@ 10,10 say "Do you want the Hit List Report" get ans
read
if ans = 'Y' .or. ans = 'y' then
   do htlstrep
endif
use tempsf
wait "Check printer and strike any key when ready"
report form sf120rep to print
eject
delete all
pack
close all
do 1342irep
return

*eof BLD_DPEN.PRG
```

```
*CALJUL.UTL
*Author:        B. A. Whitehouse
*Date:          23 July 1987
*Purpose:       Convert system date to julian date format.
*Called By:     BLD_DPEN.PG
*Calls:         None
*Input:         Calender date from within the system.
*Output:        Julian - Julian date in DDDD format.

set talk off

do while .t.
   clear

*  Input date to be changed.

   Day = DAY(DATE())
   Month = MONTH(DATE())
   Year = YEAR(DATE())
   public julian

   Yr = mod(Year,100)
   leap = mod(yr,4)

   do case

      case Month = 1.0
         julian = (mod(Yr,10)*1000) + day
      case Month = 2.0
         julian = (mod(Yr,10)*1000) + 30 + day
      case Month = 3.0
         julian = (mod(Yr,10)*1000) + 58 + day
      case Month = 4.0
         julian = (mod(Yr,10)*1000) + 89 + day
      case Month = 5.0
         julian = (mod(Yr,10)*1000) + 119 + day
      case Month = 6.0
         julian = (mod(Yr,10)*1000) + 150 + day
      case Month = 7.0
         julian = (mod(Yr,10)*1000) + 180 + day
      case Month = 8.0
         julian = (mod(Yr,10)*1000) + 211 + day
      case Month = 9.0
         julian = (mod(Yr,10)*1000) + 242 + day
      case Month = 10.0
         julain = (mod(Yr,10)*1000) + 272 + day
      case Month = 11.0
         julain = (mod(Yr,10)*1000) + 303 + day
      case Month = 12.0
         julian = (mod(Yr,10)*1000) + 334 + day
      otherwise
         @ 15,10 say [It's screwed!]
```

```
            return
      endcase

      store 1 to leapday

      if leap = 0.0 .and. Month > 2.0 then
         julian = leapday + julian
      endif

return
enddo

*eof CALJUL.UTL
```

```
*CHKJUL.PRG
*Author:        B. A. Whitehouse
*Date:          9 August 1987
*Purpose:       To recalculate the julian date in DATE_TERM
*               after adding days to the current termination
*               date.
*I/O Files:     None
*Called by:     BLD_DPEN.PRG
*Calls:         None

PARAMETERS Mdate_t

Year = YEAR(DATE())
Yr = MOD(Year,100)
leap = MOD(Yr,4)

if leap = 0 then
   if MOD(Mdate_t,1000) > 366 then
      Mdate_t = (Mdate_t - 366) + 1000
   endif
else
   if MOD(Mdate_t,1000) > 365 then
      Mdate_t = (Mdate_t - 365) + 1000
   endif
endif

? Mdate_t

*eof CHKJUL.PRG
```

```
*HTLSTREP.PRG
*Author:        B. A. Whitehouse
*Date:          25 July 1987
*Purpose:       This will print out the "HitList."  The Hit
*               List is a list of those items pending
*               disposition and
*               have not yet reached their termination date.
*I/O Files:     PEND-D.DBF, PROPERTY.DBF, NID_DPEN.NDX,
*               RPT_DPEN.NDX, NID_PROP.NDX, HITLIST.DBF,
*               HITLIST.FRM
*Calls:         None
*Called by:     DISPOSIT.PRG, BLD_DPEN.PRG

clear
set talk off

use pend-d index nid_dpen
select 2
use property index nid_prop
select 1

join with property to hitlist for nid_no = B->nid_no fields
nid_no,date_term,noun_name,qty_e,prop_code,cond_c_e,-
dept_code,rpt_no

close all

use hitlist
go top

@ 10,10 say "Is check printer set up and "
wait

report form hitlist for date_term>julian to print
eject
close all
erase hitlist.dbf
return

*eof HTLSTREP.PRG
```

```
*1342IREP.PRG
*Author:      B. A. Whitehouse
*Date:        28 August 1987
*Purpose:     Prints a form DD1342 with inputs from
*             all necessary dbfs with the same NID no.
*I/O:         PROPERTY.DBF, PLANT.DBF, MFG_INFO.DBF, *
   INV_INFO.DBF, PURCHASE.DBF; PRNT1342.FMT
*Called By:   BLD_DPEN.PRG, (DISPOSIT.PRG), (PMPS.PRG)
*Calls:       None


clear
set talk off
erase temp_1.dbf
erase temp_2.dbf
erase temp_3.dbf
erase temp_A.dbf
erase nid_A.ndx

use temp_nid
if RECCOUNT() = 0 then
   use
   return
endif
use

use property
select 2
use plant
select 1
join with plant to temp_1 for nid_no = plant -> nid_no
close all


use temp_1
select 2
use mfg_info
select 1
join with mfg_info to temp_2 for nid_no = mfg_info -> nid_no
close all


use temp_2
select 2
use inv_info
select 1
join with inv_info to temp_3 for nid_no = inv_info -> nid_no
close all


use temp_3
select 2
```

```
use purchase
select 1
join with purchase to temp_A for nid_no=purchase->nid_no
fields nid_no,julian,stock_no,cost,dept_code,mfg_year,-
mfg_name,mf_code,mfg_mod_no,comod_code,type_code,-
non_avl_no,pwr_code,length,width,height,weight,noun_name,-
po_no,mfg_ser_no,locate

close all
use temp_nid
select 2
use temp_A
index on nid_no to nid_A

do while .not. EOF()
   select 2
   find &temp_nid->nid_no
   clear
   if FOUND()
      set device to print
      do prnt1342.fmt
      eject
   endif
   select 1
   skip
enddo

set device to screen
select 1
go top
delete all
pack
close all databases

erase temp_1.dbf
erase temp_2.dbf
erase temp_3.dbf
erase temp_A.dbf
erase nid_A.ndx
wait to continue
clear

return


*Eof  1342IREF.PRG
```

```
*SF120REP.PRG
*Author:         B. A. Whitehouse
*Date:           8 August 1987
*Purpose:        To append information to a temporary file
*                (TEMPSF.DBF) which will later be used to
*                create a report called SF120REP used to type
*                up the SF 120.
*I/O Files:      TEMPSF.DBF
*Called by:      BLD_DPEN.PRG
*Calls:          None

clear

*memory variables
Mqty = 0
Mrpt_num = space(11)
Mname = space(30)

store nid_no to NID
store qty_e to Mqty
store rpt_no to Mrpt_num
select 1
store noun_name to Mname

select 9
use tempsf
append blank
replace nid_no with NID
replace noun_name with Mname
replace cost with mfg_info->cost
replace qty_e with Mqty
replace rpt_no with Mrpt_num
use

return

*eof SF120REP.PRG
```

```
*DISP_MGT.PRG
*Author:          B. A. Whitehouse
*Date:            8 July 1987
*Purpose:         This module allows the user to manage or
*                 change status of records on the
*                 PENDING-D FILE, (done when processing DARIC/
*                 DIPEC_INST, or DATE_TERM requests) or
*                 modify records on the disposition file.
*I/O Files:       None
*Called By:       PMPS.PRG
*Calls:           MGTSTAT.PRG, MODREC.PRG

clear
set talk off
store "  " to choice

do while .t.
    @ 2,0 to 21,79 double
    @ 3,25 say [Dispositions Management Menu]
    @ 4,1 to 4,78 double
    @  7,20 say [  Task                              Task Code]
    @  9,20 say [Manage Status Pending-D File           1]
    @ 10,20 say [Modify Record                          2]
    @ 12,20 say [Return to PMPS Main Menu               3]
?
?
?
    wait "      Enter your choice (Task Code):  " to choice

    do case
        case choice = '1'
            do mgtstat

        case choice = '2'
            do modrec

        case choice = '3'
            clear
            return

        otherwise
            loop

    endcase
close all
enddo

*eof DISP_MGT.PRG
```

```
*MGTSTAT.PRG
*Author:                 B. A. Whitehouse
*Date:                   30 July 1987
*Purpose:                This module allows the user to update
*                        status on the Pending Dispositions
*                        file according to instructions
*                        received from DARIC and/or DIPEC.
*I/O Files:              PEND-D.DBF, NID_DPEN.NDX, RPT_DPEN.NDX
*Called By:              DISP_MGT.PRG
*Calls:                  INPUTINS.PRG, CHKPHASE.PRG,
GETRPT.PRG,
*                        GETNID.PRG

clear
set talk off
store "  " to choice
do caljul.utl

clear
rpt_num = space(11)
ans    = "Y"

do while .t.
   @ 2,0 to 21,79 double
   @ 3,23 say [Record Processing & Status Update]
   @ 4,1 to 4,78 double
   @  7,18 say [   Task                          Task Code]
   @  9,18 say [Received DARIC Instructions          1]
   @ 10,18 say [Received DIPEC Instructions          2]
   @ 11,18 say [Process Termination Date             3]
   @ 13,18 say [Return to Disposition Mgt. Menu      4]
?
?
?
   wait "      Enter your choice (Task Code):  " to choice
   clear

   do case
      case choice = '1'
         do while .t.
            do getrpt
            @ 7,10 say [Is report number &MRPT correct?] get
ans
            read
            if ans = 'Y' .or. ans = 'y' then
               exit
            endif
         enddo

         use pend-d index rpt_dpen,nid_dpen
         find &MRPT
         do while .not. EOF() .and. rpt_no = MRPT
```

```
            do inputin
            if date_term <= julian .and. phase = "R" then
               phase = "DL"
            endif
            skip
         enddo
         close all

       case choice = '2'
         do getnid
         use pend-d index nid_dpen,rpt_dpen
         find &NID
         if FOUND() then
            do inputin
            if date_term <= julian .and. phase = "R" then
               phase = "DL"
            endif
         endif
         close all

      case choice = '3'
         use pend-d index nid_dpen,rpt_dpen
         go top
         do while .not. EOF()
            if date_term <= julian .and. phase = "R" then
               phase = "DL"
            endif
            skip
         enddo
         close all

      case choice = '4'
         use pend-d index nid_dpen,rpt_dpen
         do chkphase
         close all
         clear
         return

      otherwise
         @ 13,18 say [Enter one of the Task Codes.]
         loop
   endcase
  clear
enddo

*eof MGTSTAT.PRG
```

```
*GETRPT.PRG
*Author:        B. A. Whitehouse
*Date:          11 August 1987
*Purpose:       Solicits REPORT number from the user, shows the
*               user a template then checks for proper length
*               and alphabetic characters.  It continues to
*               loop until a proper REPORT number is input.
*I/O:           MRPT (Public Memory Variable)
*Called By:
*Calls:         None


clear
set talk off
public MRPT

MRPT = "N62271-     "
store .t. to check

do while check

@12,05 say [Enter the Report number followed by <RTN>:]
@12,47 get MRPT picture 'A99999A9999'
read
    if len(rtrim(MRPT)) < 11
            store .t. to check
      @14,10 say [The Report Number you input was too short;]

      @15,10 say [please check and try again.]
      @16,10 say []
            wait to continue
            clear
      else
            store .f. to check
            clear
    endif
enddo


*eof GETRPT.PRG
```

```
*INPUTIN.PRG
*Author:        B. A. Whitehouse
*Date:          31 July 1987
*Purpose:       This module allows the user to update the
*               information contained in the pending
*               dispositiion record according to the
*               instructions received from DARIC and/or DIPEC.
*I/O Files:     PEND-D.DBF, NID_DPEN.NDX, RPT_DPEN.NDX
*Called By:     MGTSTAT.PRG
*Calls:         None

clear
set talk off

store date_term to inst_d_date
store phase to inst_act
store nid_no to tempnid

@ 7,7 say [For NID number &tempnid ...]
@ 9,7 say [Enter ARD from DIPEC/DARIC instruction.]
@ 10,10 say [Julian date format (YDDD)] get inst_d_date
picture
'9999'
@ 12,7 say [Enter instructed action (PHASE = 'DL' or 'DT')]
get
inst_act
read

replace date_term with inst_d_date
replace phase with inst_act
return

*eof INPUTIN.PRG
```

```
*CHKPHASE.PRG
*Author:         B. A. Whitehouse
*Date:           1 August 1987
*Purpose:        This module uses the Pending Dispositions
*                (PEND-D) file and prints a DD form or a
*                report according the code appearing in the
*                PHASE field.  A phase of 'DL' will cause a
*                1348-1 to be printed and set phase = 'DLH'.
*                A phase code of 'DT' will cause a Transfer
*                Report to be created and set phase = 'DTH'.
*                Finially, if there is a 'DLH' or 'DTH' in
*                the phase code a Hold Report will be
*                created.
*I/O Files:      PEND-D.DBF, PROPERTY.DBF, MFG_INFO.DBF,
*                NID_DPEN.NDX, RPT_DPEN.NDX, NID_PROP.NDX,
*                NID_MFG.NDX
*Called By:      MGTSTAT.PRG
*Calls:          DD1348-1.PRG, TRANSREP.PRG

set talk off
clear
go top
select 2
use property index nid_prop
select 1
set relation to nid_no into property
select 3
use mfg_info index nid_mfg
select 2
set relation to nid_no into mfg_info
select 1

*       When it can be determined the exact format
*       for each 1348-1, this should be coded to print out
*       the 1348-1.  In the mean time it will only print a
*       report containing a list of those items on which a
*       1348-1 should be created.

do transrep
?
WAIT [For hard copy press <SHIFT><PRINT>, otherwise any key
to
continue.]
do DD1348-1
?
WAIT [For hard copy press <SHIFT><PRINT>, otherwise any key
to
continue.]
go top
do holdrep
?
WAIT [For hard copy press <SHIFT><PRINT>, otherwise any key
to continue.]
close all
return

*eof CHKPHASE.PRG
```

```
*TRANSREP.PRG
*Author:         B. A. Whitehouse
*Date:           12 August 1987
*Purpose:        To print a report of those items which have
*                been designated for transfer to some other
*                military installation.
*I/O Files:      PROPERTY.DBF, PEND-D.DBF, MFG_INFO.DBF
*Calls:          None
*Called By:      CHKPHASE.PRG

set talk off
clear
store 10 to lineno

go top
@ 0,0
@ 2,30 say "Transfer Report"
@ 3,15 say "Items being transfered to other military bases"
@ 5,10 say DATE()
@ 7,62 say "Date"
@ 7,69 say "Dept"
@ 8,4 say "P/C"
@ 8,8 say "NID Number"
@ 8,20 say "      Item Name"
@ 8,50 say "Report No."
@ 8,63 say "Term"
@ 8,69 say "Code"

do while .not. EOF()
   if UPPER(phase) = "DT " .and. .not. DELETED()
      @ lineno,4 say property->prop_code
      @ lineno,8 say nid_no
      @ lineno,22 say property->noun_name
      @ lineno,50 say rpt_no
      @ lineno,63 say date_term
      @ lineno,69 say property->dept_code
      store lineno + 1 to lineno
      replace phase with "DTH"
   endif
   skip
enddo
return

*eof TRANSREP.PRG
```

```
*DD1348-1.PRG
*Author:        B. A. Whitehouse
*Date:          12 August 1987
*Purpose:       To print a report of those items which have
*               been designated for disposition locally.
*               A DD 1348-1 itself should be printed.
*I/O Files:     PROPERTY.DBF, PEND-D.DBF, MFG_INFO.DBF
*Calls:         None
*Called By:     CHKPHASE.PRG

clear
store 10 to lineno

go top
@ 0,0
@ 2,23 say "Items For Local Disposition"
@ 3,19 SAY "These items need a DD 1348-1 typed"
@ 5,10 say DATE()
@ 7,4 say "NID Number"
@ 7,17 say "P/C"
@ 7,21 say "Stock No."
@ 7,28 say "     Item Name"
@ 7,58 say "MFR Ser. No."
@ 8,46 say "MFR Model No."

do while .not. EOF()
   if UPPER(phase) = "DL " .and. .not. DELETED() .and.
date_term
< julian
      @ lineno,4 say nid_no
      @ lineno,18 say property->prop_code
      @ lineno,22 say mfg_info->stock_no
      @ lineno,29 say property->noun_name
      @ lineno,58 say property->mfg_ser_no
      store lineno + 1 to lineno
      @ lineno,47 say property->mfg_mod_no
      store lineno + 1 to lineno
      replace phase with "DLH"
   endif
   skip
enddo
return

*eof DD1348-1.PRG
```

```
*HOLDREP.PRG
*Author:        B. A. Whitehouse
*Date:          13 August 1987
*Purpose:       To print a report of those items which have
*               been placed on hold awaiting conformation of
*               receipt by those who are to receive the items.
*I/O Files:     PROPERTY.DBF, PEND-D.DBF, MFG_INFO.DBF
*Calls:         None
*Called By:     CHKPHASE.PRG

set talk off
clear
store 10 to lineno

go top
@ 0,0
@ 2,35 say "Hold Report"
@ 3,2 say "Records of items which have been sent out now
awaiting a signed 1348-1 or 1149"
@ 5,10 say DATE()
@ 7,55 say "Date"
@ 7,61 say "Dept"
@ 8,6 say "P/C"
@ 8,11 say "NID Number"
@ 8,24 say "     Item Name"
@ 8,55 say "Term"
@ 8,61 say "Code"

do while .not. EOF()
   if UPPER(phase) = "DLH" .or. UPPER(phase) = "DTH" .and.
.not. DELETED()
      @ lineno,7 say property->prop_code
      @ lineno,11 say nid_no
      @ lineno,24 say property->noun_name
      @ lineno,55 say date_term
      @ lineno,61 say property->dept_code
      store lineno + 1 to lineno
   endif
   skip
enddo
return

*eof HOLDREP.PRG
```

```
*MODREC.PRG
*Author:      B. A. Whitehouse
*Date:        3 August 1987
*Purpose:     To allow the user to select from a menu the
*             various programs which will allow him to delete
*             a record, update or correct data, print a 1342.
*I/O Files:   None
*Called by:   DISP_MGT.PRG
*Calls:       DLET_REC.PRG, FIX_DATA.PRG, PRNT1342.PRG,
*             REUTILIZ.PRG

clear
set talk off
store "  " to choice
text

                    MODIFICATION OF RECORDS

The following screen will allow you to delete records, make
corrections to data, print a 1342, or process an item for
reutilization.  Processing the signed 1149 (reciept) and
1348-1
from DRMO are the same thing, you no longer have to carry
these
items on your books.

endtext
wait to cont
clear
do while .t.
   @ 2,0 to 21,79 double
   @ 3,30 say [Modification Menu]
   @ 4,1 to 4,78 double
   @ 7,20 say [      Task                          Task Code]
   @ 9,20 say [Process signed DD 1149 reciept         1]
   @ 10,20 say [Process signed DD 1348-1               2]
   @ 11,20 say [Make Corrections to Data               3]
   @ 12,20 say [Print DD 1342                          4]
   @ 13,20 say [Delete a Record                        5]
   @ 14,20 say [Process Item for Reutilization         6]
   @ 16,20 say [Return to Disposition Mgt. Menu        7]

?
?
?
   wait "     Enter your choice (Task Code):  " to choice

   do case
     case choice = '1'
        do dlet_rec

     case choice = '2'
```

```
            do dlet_rec

         case choice = '3'
            do fix_data

         case choice = '4'
            do prnt1342

         case choice = '5'
            do dlet_rec

         case choice = '6'
            do reutiliz

         case choice = '7'
            clear
            return

         otherwise
            @ 17,30 say [ Enter one of the choices only.]
            loop

        endcase
    close all
    enddo

    *eof MODREC.PRG
```

```
*DLET_REC.PRG
*Author:        B. A. Whitehouse
*Date:          3 August 1987
*Purpose:       To allow the user to select a record to from
*               the database by inputing a NID number or
*               document number and then deleting all
*               associated fields from each relation.
*I/O Files:     PROPERTY.DBF, PLANT.DBF, PENDING.DBF,
*               PEND-D.DBF, MFG_INFO.DBF, PURCHASE.DBF,
*               INV_INFO.DBF, NID_PROP.NDX, NIDPLANT.NDX,
*               NIDPEND.NDX, NID_DPEN.NDX, RPT_DPEN.NDX,
*               NID_MFG.NDX, NID_PURC.NDX, NID_INV.NDX
*Called by:     MODREC.PRG
*Calls:         GETNID.PRG

clear
ans    = 'Y'
nid    = space(12)
sure   = 'N'

do while .t.
   @ 5,7 say [Do you have a NID number for the record to be
deleted?]
   @ 6,7 say [    'Y' for yes or 'R' to return to Main Menu]
get
ans
   read
?
?
   if ans = 'R' .or. ans = 'r' then
      clear
      return
   endif

   if ans = 'Y' .or. ans = 'y' then
      clear
      do getnid
      @8,10 say [Are you sure you want to delete this
record?] get sure
      read
      if sure = 'Y' .or. sure = 'y' then
         clear
         @ 5,7 say [Deleting Record, &NID]
         use property index nid_prop
         find &NID
         if .not. FOUND() then
            @ 7,5 say [Record not found.]
            ?
            wait
         else
            if prop_code = 'P' .or. prop_code = 'p' then
               select 2
```

```
                    use plant index nidplant
                    find &NID
                    delete
                    pack
                    use
                    select 1
                    delete
                    pack
                 else
                    delete
                    pack
                    use pending index nidpend
                    find &NID
                    delete
                    pack
                 endif
                    use pend-d index nid_dpen
                    find &NID
                    if FOUND() then
                       delete
                       pack
                    endif
                    use mfg_info index nid_mfg
                    find &NID
                    delete
                    pack
                    use purchase index nid_purc
                    find &NID
                    delete
                    pack
                    use inv_info index nid_inv
                    find &NID
                    delete
                    pack
                    use
                    return
              endif
              clear
           else
              clear
              return
           endif
        else
           clear
           @ 9,5 say [Enter "Y" or "R" for Yes or Return.]
           @ 10,5 say [If you haven't got a NID Number, use the
     Modify Record part]
           @ 11,5 say [of this System to browse the files but now,
     use]
           @ 12,7 say [ an "R" to return to the previous menu.]
        endif
     enddo
```

```
clear
return

*eof DLET_REC.PRG
```

```
*FIX_DATA.PRG
*Author:                B. A. Whitehouse
*Date:                  11 August 1987
*Purpose:               To allow the user to make corrections
*                       to the records in the logical Pending
*                       Dispostions file.
*I/O Files:             None
*Called by:             MODREC.PRG
*Calls:                 FIXBYNID.PRG, FIXBYRPT.PRG

clear
set talk off
store "  " to choice
text

                        MODIFICATION OF RECORDS

You will find it far easier to locate and correct existing
records if you use the NID Number to locate them.  This is of
course not always possible.  The next best thing is to use
the Report Number and then list each record having that
report number untill you find the one you are looking for.

endtext
wait to cont
clear
do while .t.
   @ 2,0 to 21,79 double
   @ 3,21 say [Corrections to Pending Disposition File]
   @ 4,1 to 4,78 double
   @ 7,21 say [          Task                Task Code]
   @ 9,21 say [Using a NID Number                   1]
   @ 10,21 say [Using a Report Number                2]
   @ 12,21 say [Return to Modification Menu          3]

?
?
?
   wait "     Enter your choice (Task Code):  " to choice

   do case
     case choice = '1'
        do fixbynid

     case choice = '2'
        do fixbyrpt

     case choice = '3'
        clear
        return

     otherwise
```

```
        @ 17,30 say [ Enter one of the choices only.]
        loop

    endcase
enddo

*eof FIX_DATA.PRG
```

```
*FIXBYNID.PRG
*Author:        B. A. Whitehouse
*Date:          11 August 1987
*Purpose:       To allow the user to make corrections to the
*               records in the logical Pending Dispositions
*               file using the NID number to find them.
*I/O Files:     PEND-D.DBF, PROPERTY.DBF, PENDING.DBF,
*               MFG_INFO.DBF, NID_DPEN.NDX, RPT_DPEN.NDX,
*               NID_PROP.NDX, NIDPEND.NDX, NID_MFG.NDX
*Called by:     FIX_DATA.PRG
*Calls:         GETNID.PRG

clear
set talk off
clear
again = .t.
answ  = space(1)

text

Make whatever changes are appropriate to the record which
appears on the screen.  Be sure to varify that the NID Number
is the one you intended to get.  Upon entering the last field
(Status) the record will be updated.

endtext

select 4
use pend-d index nid_dpen,rpt_dpen
select 1
use property index nid_prop
select 4
set relation to nid_no into property
select 3
use pending index nidpend
select 1
set relation to nid_no into pending
select 5
use mfg_info index nid_mfg
select 3
set relation to nid_no into mfg_info
select 4
wait

   go top
   do getnid
   find &NID

   if FOUND() then
      cond_cd      = cond_c_e
      date_ent     = date_ent_d
      term_dt      = date_term
```

```
phase_cd     = phase
poc          = poc_n
poc_ph       = poc_p_no
qty          = qty_e
rept_num     = rpt_no

select 3
status       = pend_stat

select 1
mfr_name    = mfg_name
serial_no   = mfg_ser_no
n_name      = noun_name
model_no    = mfg_mod_no
prop_cd     = prop_code
dept        = dept_code

select 5
mfr_yr      = mfg_year
stock_num   = stock_no
type_cd     = type_code
price       = cost

@ 0,25 say [Disposition Record for NID ] get NID
@  2,3 say [Manufacture's Name:] get mfr_name
@  3,3 say [Mfr's Serial Number:] get serial_no
@  4,3 say [Item Noun Name:] get n_name
@  5,3 say [Mfr's Model Number:] get model_no
@  6,3 say [Property Code:] get prop_cd
@  7,3 say [Department Code:] get dept
@  8,3 say [Year Manufactured:] get mfr_yr
@  9,3 say [Stock Number:] get stock_num
@ 10,3 say [Type Code:] get type_cd
@ 11,3 say [Cost at purchase:] get price
@ 12,3 say [Condition Code:] get cond_cd
@ 13,3 say [Date Entered Dispositions:] get date_ent
@ 14,3 say [Termination Date:] get term_dt
@ 15,3 say [Phase code:] get phase_cd
@ 16,3 say [Point of Contact;] get poc
@ 17,3 say [POC Phone Number:] get poc_ph
@ 18,3 say [Quatity Excess:] get qty
@ 19,3 say [Report Number:] get rept_num
@ 20,3 say [Status of Record:] get status

read
select 1
replace mfg_name with mfr_name
replace mfg_ser_no with serial_no
replace noun_name with n_name
replace mfg_mod_no with model_no
replace prop_code with prop_cd
replace dept_code with dept
```

```
      select 5
      replace mfg_year with mfr_yr
      replace stock_no with stock_num
      replace type_code with type_cd
      replace cost with price
      select 3
      replace pend_stat with status
      select 4
      replace cond_c_e with cond_cd
      replace date_ent_d with date_ent
      replace date_term with term_dt
      replace phase with phase_cd
      replace poc_n with poc
      replace poc_p_no with poc_ph
      replace qty_e with qty
      replace rpt_no with rept_num

      clear
   else
      @ 10,10 say [Record not found.]
      ?
      wait
   endif
close all
clear
return

*eof FIXBYNID.PRG
```

```
*FIXBYRPT.PRG
*Author:         B. A. Whitehouse
*Date:           11 August 1987
*Purpose:        To allow the user to make corrections to the
*                record in the logical Pending Dispositions
*                file using the Report Number to find them.
*I/O Files:      PEND-D.DBF, PROPERTY.DBF, PENDING.DBF,
*                MFG_INFO.DBF, RPT_DPEN.NDX, NID_DPEN.DBF,
*                NID_PROP.NDX, NIDPEND.NDX, NID_MFG.NDX
*Called by:      FIX_DATA.PRG
*Calls:          GETRPT.PRG

clear
set talk off
clear
again = .t.
answ  = space(1)

text

Make whatever changes are appropriate to the record which
appears on the screen.  Be sure to varify that the Report
Number is the one you intended to get.  Upon entering the
last field (status) the record will be updated.
(YOU DON'T HAVE TO CHANGE ANYTHING)


endtext

select 4
use pend-d index rpt_dpen,nid_dpen
select 1
use property index nid_prop
select 4
set relation to nid_no into property
select 3
use pending index nidpend
select 1
set relation to nid_no into pending
select 5
use mfg_info index nid_mfg
select 3
set relation to nid_no into mfg_info
select 4
wait

do while .t.
   go top
   do getrpt
   find &MRPT
   if .not. FOUND() then
      @ 10,10 say [Record not found]
      wait
   endif

   do while rpt_no = MRPT
      select 4
      NID            = nid_no
      cond_cd        = cond_c_e
      date_ent       = date_ent_d
      term_dt        = date_term
      phase_cd       = phase
      poc            = poc_n
      poc_ph         = poc_p_no
      qty            = qty_e
      rept_num       = rpt_no

      select 3
      status         = pend_stat

      select 1
      mfr_name      = mfg_name
      serial_no     = mfg_ser_no
      n_name        = noun_name
      model_no      = mfg_mod_no
      prop_cd       = prop_code
      dept          = dept_code

      select 5
      mfr_yr        = mfg_year
      stock_num     = stock_no
      type_cd       = type_code
      price         = cost

      select 4

      do fixby.fmt
      read

      @ 19,5 say [Have you changed this record? (Y/N)] get
answ
      read
      if answ = 'Y' .or. answ = 'y' then
         select 1
         replace mfg_name with mfr_name
         replace mfg_ser_no with serial_no
         replace noun_name with n_name
         replace mfg_mod_no with model_no
         replace prop_code with prop_cd
         replace dept_code with dept
         select 5
         replace mfg_year with mfr_yr
         replace stock_no with stock_num
         replace type_code with type_cd
         replace cost with price
         select 3
         replace pend_stat with status
         select 4
         replace cond_c_e with cond_cd
         replace date_ent_d with date_ent
         replace date_term with term_dt
         replace phase with phase_cd
         replace poc_n with poc
         replace poc_p_no with poc_ph
         replace qty_e with qty
         replace rpt_no with rept_num

         clear
         select 4
         @ 10,10 say [Would you like to do anohter which may
have]
         @ 11,12 say [the same report number? (Y/N)] get answ

         read
         if answ = 'Y' .or. answ = 'y' then
            skip
            clear
         else
            exit
         endif
      else
         select 4
         clear
         @ 10,10 say [Would you like to look at another
record
which may]
         @ 11,10 say [have the same report number? (Y/N)] get

answ
         read
         clear
         if answ = 'Y' .or. answ = 'y' then
            skip
         else
            exit
         endif
      endif
   enddo
   select 4
   clear
   @ 10,10 say [Want to look at different report #? (Y/N)]
get answ
   read
   if answ = 'N' .or. answ = 'n' then
      clear
      exit
   endif
enddo
close all
clear
return

*eof FIXBYRPT.PRG
```

```
*REUTILIZ.PRG
*Author:        B. A. Whitehouse
*Date:          3 August 1987
*Purpose:       This module allows the user to delete items
*               identified for reutilization at this command
*               from the pending disposition, to change the
*               department code, and location of the new
*               owner.
*I/O Files:     PROPERTY.DBF, INV_INFO.DBF, PEND-D.DBF,
*               PENDING.DBF, NID_PROP.NDX, NID_INV.NDX,
*               NIDPEND.NDX, NDI_DPEN.NDX, RPT_DPEN.NDX
*Called by:     MODREC.PRG
*Calls:         GETNID.PRG, GETRPT.PRG

clear
ans    = "Y"
nid    = space(12)
dept   = space(4)
Mbldg  = space(4)
Mroom  = space(4)

do while .t.
   @ 7,5 say [Do you have a NID number for item to be
reused?]
   @ 8,5 say [   (Y/N) or R to return to Main Menu] get ans
   read
   if ans = "R" .or. ans = "r" then
      clear
      close all
      return
   endif
   if ans = "Y" .or. ans = "y" then
      clear
      do getnid
      select 1
      use property index nid_prop
      select 2
      use inv_info index nid_inv
      select 1
      set relation to nid_no into inv_info
      find &NID
      if FOUND() then
         @ 5,5 say [Enter new Department Code:] get dept
         @ 7,5 say [Enter new Building Number:] get Mbldg
         @ 9,5 say [Enter new Room Number:] get Mroom
         read
         replace dept_code with dept
         select 2
         replace locate with Mbldg-Mroom
         select 4
         use pend-d index nid_dpen
         find &NID
```

```
          delete
          pack
          select 3
          use pending index nidpend
          find &NID
          replace pend_stat with "R"
          clear
          close all
       else
          @ 13,10 say [NID does not exist in Property file.]
          wait
          clear
       endif
    else
       if ans = "N" .or. ans = "n" then
          clear
          do getrpt
          select 4
          use pend-d index rpt_dpen,nid_dpen
          find &MRPT
          if FOUND() then
             clear
             do while .not. EOF() .and. rpt_no = MRPT
                display
                @ 15,10 say [Is this the record you are
looking
for? (Y/N)] get ans
                read
                if ans = 'Y' .or. ans = 'y' then
                   replace NID with nid_no
                   delete
                   pack
                   use property index nid_prop
                   select 2
                   use inv_info index nid_inv
                   select 1
                   set relation to nid_no into inv_info
                   find &NID
                   @ 5,5 say [Enter new Department Code:] get
dept
                   @ 7,5 say [Enter new Building Number:] get
Mbldg
                   @ 9,5 say [Enter new Room Number:] get
Mroom
                   replace dept_code with dept
                   select 2
                   replace locate with Mbldg-Mroom
                   select 3
                   use pending index nidpend
                   find &NID
                   replace pend_stat with "R"
                   use
```

```
                    select 4
                    clear
                    skip
                 else
                    if ans = 'N' .or. ans = 'n' then
                       skip
                       clear
                    else
                       clear
                       @ 21,10 say [YES or NO (Y/N)]
                    endif
                 endif
              enddo
              clear
              @ 5,10 say [You have reached the end of the file]

              @ 6,10 say [and/or you have found what you are
looking for.]
              ?
              wait
              close all
              exit
           else
              @ 9,5 say [Record does not exist in Pending
Dispositions]
              wait
              close all
              return
           endif
        else
           @ 9,5 say [Enter "Y" or "N" for Yes or No.]
        endif
     endif
enddo

*eof REUTILIZ.PRG
```

APPENDIX D

USER'S QUICK REFERENCE GUIDE

The users' Quick Reference is intended to assist MCD employees in using the PMPS automated prototype described in this thesis. It is assumed the individual is already familiar with Navy-wide as well as local practices and procedures used in accounting for plant and minor property. The Quick Reference is not meant to train the individuals in these standard operating procedures, nor in the use of the dBase III Plus data base management system and its command language.

This system is only a prototype and not intended for use as a production product. There are instances throughout the system that what can and cannot happen presently is less than desirable; for instance:

1. No single command is available that will consistently allow the user to quit and escape from what is being done, and be assured the data base will remain in good shape.

2. There are many cases where, once a series of input screens start to appear on the monitor, the user must work through each screen. If the user has nothing worthwhile to input, the data base will be loaded with insignificant data or have blank records. This will create work for the data base administrator.

3. There are very few routines to prevent illegal and/or insignificant data from being entered.

Plant and Minor Property System
Quick Reference Guide
Index

QUICK REFERENCE

1. **PROGRAM NAME:** PMPS.PRG (Plant and Minor Property System)

2. **PROMPT:** The PMPS Main Menu screen (Figure D.1).

3. **PURPOSE:** Accessing the major sub-sections of the PMPS, in order to enter, update, and close out plant and minor property records.

4. **TO RUN:** Initiate dBase III Plus by having the software loaded into the computer and typing "DBASE". Select the appropriate default drive using the "ASSIST" command. At the dBase III Plus "." prompt, type "DO PMPS" <RTN>. The screen shown in Figure D.1 will appear.

5. **TO USE:** Select the desired option from the menu by typing its Task Code number.

    1. Acquisitions. This option will bring up ACQUISIT.PRG (page 189), for entering data about new equipment.

    2. Process Disposition Transaction. Brings up INPUT_TR.PRG (page 213), initiating the disposition process for items now determined to be excess.

    3. Disposition Management. Brings up DISP_MGT.PRG (page 221), which allows the user to complete, update, or modify records of equipment pending disposition.

    4. Ad Hoc Requests. The user may write short programs (macros) that do specific tasks performed often, if desired. These may be called up via this selection.

6. **TO GET OUT:** Enter the Task Code number "5" from the PMPS Main Menu, to select "Quit" and return to the dBase III Plus level.

7. **PROBLEMS/CAVEATS:**

    a. Ideally dBase III Plus and this system will all be resident on a hard disk under one directory.

    b. You may enter any character or press any key except the <ESC> key. If the <ESC> key is pressed, the following message will appear:

```
***INTERRUPTED***
Called from - A:pmps.prg

        Cancel, Ignore, Suspend?(C,I, or S)
```

Type "I" for ignore. If you enter "C" you must begin again from dBase III Plus (see TO RUN:). If "S" is entered you must type "CANCEL" at the "." prompt and start over.

c. No macro commands exist, for the Ad Hoc Request category. As the user becomes more familiar with the dBase III Plus command set, these can be built and inserted into an ad hoc menu.

```
                              Main Menu
                  Task                                 Task Code

                  Acquisitions                             1
                  Process Disposition Transaction          2
                  Disposition Management                   3
                  Ad Hoc Requests                          4
                  Quit                                     5
          Enter Task Code:
```

Figure D.1 PMPS Main Menu

QUICK REFERENCE

1. **PROGRAM NAME:** ACQUISIT.PRG (Acquisitions)

2. **PROMPT:** The Acquisition Main Menu screen (Figure D.2).

3. **PURPOSE:** To select subprograms related to acquisition of new property.

4. **TO RUN:** Type the Task Code number "1" at the PMPS Main Menu (Figure D.1). The screen shown in Figure D.2 will appear.

5. **TO USE:** Select the desired option from the menu by typing its Task Code number.

ACQUISITIONS MAIN MENU

Task	Task Code
Notice of Pending Receipt	1
Equipment Receipt	2
Reconcile 1342	3
Print 1342	4
Modify 1342	5
Inventory Files	6
Pending Files	7
Return to Main Menu	8

Enter Task Code:

Figure D.2 Menu resulting from selection of Main Menu Task Code "1"

1. Notice of Pending Receipt. Brings up NT_PEN_R.PRG (page 191), which is used when adding new property records ordered but not yet on hand.

2. Equipment Receipt. Brings up EQ_RECPT.PRG (page 195), used when property pending receipt is received.

3. Reconcile 1342. Brings up MAKE_1342.PRG (page 198), used to enter data missing from the DD 1342.

4. Print 1342. Brings up PRNT1342.PRG (page 201), used to print a hard copy of specified DD 1342 information on blank paper.

5. Modify 1342. Brings up VIEW1342.PRG (page 203), used to review and make changes to the data base concerning the Form DD 1342.

6. Inventory Files. Brings up INVMENU.PRG (page 205) used to query and/or print out information about inventory on hand. Generally used when conducting an inventory.

7. Pending Files. Brings up PENDMENU.PRG (page 209), used to query the Pending File.

6. TO GET OUT: Type the Task Code number "8" from the menu to return to the PMPS Main Menu.

7. PROBLEMS/CAVEATS:

a. Avoid using the <ESC> key and all of the function keys. Pressing them delays processing.

QUICK REFERENCE

1. **PROGRAM NAME:** NT_PEN_R.PRG (Notice of Pending Receipt)

2. **PROMPT:** None

3. **PURPOSE:** Provides screens allowing entry of initial data needed to complete a form DD 1342.

4. **TO RUN:** Type the Task Code number '1' at the Acquisitions Main Menu Figure D.2. The screen shown in (Figure D.3) will appear.

5. **TO USE:** The user first gets an explanatory message (Figure D.3). Press any key to continue. A screen will prompt for input.

```
**************************************************

This module lets you enter data on a new piece of
equipment that has not been received yet.  You
will be entering data on the next four screens in
order to star building your WS 1342.

**************************************************
```

Figure D.3 Screen resulting from selection of Acquisition Main Menu Task Code "1"

a. The first screen requests the NID number for this property (see GETNID.PRG, page 194).

b. Next you are asked to enter the pending status code (Figure D.4). For the PMPS program, this will always be a 'P'.

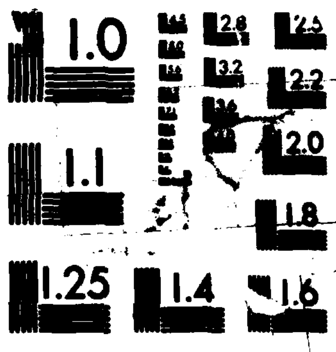
```
 Pending Status

*************************************************
* Please enter a 'P' in the Pending Status Field. *
* The next screen will appear automatically.      *
*************************************************
```

Figure D.4 First input screen in "Notice of Pending Receipt"

AD-A187 703 THE IMPLEMENTATION OF A PLANT AND MINOR PROPERTY ACCOUNTING SYSTEM FOR THE NAVAL POSTGRADUATE SCHOOL(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W T KEY ET AL. 3/3

UNCLASSIFIED SEP 87 F/G 5/1 NL


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

c. The screens shown in Figures D.5 and D.6 now follow, for entry of the rest of the required information. You simply type in the appropriate information. The arrow keys move the cursor from one input block to another. You may back up and delete typos. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

```
        Mfg Name

        Noun Name

        Mfg Model Number

        Property Code

        Department Code

   ** Press <CTRL> <END> to save the data you just entered. **
```

Figure D.5 Second input screen in "Notice of Pending Receipt"

```
        Consignor

        Date Entered

        Purchase Order Number

        Requisition Number

        ** For the Date Entered, use today's date. **

   ** Press <CTRL> <END> to save the data you just entered. **
```

Figure D.6 Third input screen in "Notice of Pending Receipt"

6. TO GET OUT: Move through the screens, entering all the information possible. The Acquisitions Main Menu will reappear after pressing <CTRL><END> for the screen shown in Figure D.6

7. **PROBLEMS/CAVEATS:**

a. An entry of a NID number already in use gives a message stating this. Press any key to continue and try again.

b. From screens in Figures D.5 and D.6 you may complete the input process by:

-striking the <CTRL><END> keys as instructed on the screen.

-filling in the last character in the last field on the screen.

-striking <ENTER> with the cursor in the last field on the screen.

c. In the first situation the screen will disappear; the information will be entered into the relations, and the process will move on.

d. In the second and third situation, the screen with the user's information will disappear and the same screen will reappear with different information or blank fields. Simply strike <PG UP> and the previous screen will display. The information you entered will reappear. Then depress the <CTRL><END> keys.

e. Correct any typing errors by backing up using the arrow keys to position yourself on the goof; then type the corrections.

QUICK REFERENCE

1. **PROGRAM NAME:** GETNID.PRG (Get the NID number)

2. **PROMPT:** Appears below.

```
Enter the NID number followed by a <RTN>: 62271-
```

3. **PURPOSE:** To enter a NID number. This program will then check this number for correctness: make sure there are 12 characters and that they are all numbers.

4. **TO RUN:** This program is automatically called from other programs.

5. **TO USE:** Enter the 12 character NID number at the prompt.

6. **TO GET OUT:** Enter a valid NID number.

7. **PROBLEMS/CAVEATS:**
   a. If the user enters an invalid NID number (i.e., less than 12 characters or with alpha characters) the program will loop until a valid NID is entered.

   b. Correct typing errors by using the arrow keys or backspace key to move the cursor.

QUICK REFERENCE

1. **PROGRAM NAME:** EQ_RECPT.PRG (Equipment Receipt)

2. **PROMPT:** None.

3. **PURPOSE:** (a) To allow the user to check property information previously entered in the data base for correctness, (b) to add to or change that information, (c) to indicate that the equipment is now on hand.

4. **TO RUN:** Type the Task Code number '2' at the Acquisitions Main Menu (Figure D.2). The screen shown in Figure D.7 will appear.

5. **TO USE:** First the user gets an explanatory message (Figure D.7). Press any key to continue. Five screens will prompt for input.

```
****************************************

At this time you will be requested to
compare the information in the Pending
File to the information on the piece
of equipment which actually arrived.

You may make necessary changes in the
highlighted fields.

****************************************
```

Figure D.7 Screen resulting from selection of Acquisition Main Menu Task Code "2"

a. The first screen requests the NID number for this property (see GETNID.PRG, page 194).

b. Next, you are asked to enter the property code with the statement, 'Enter property code:'. This is for verification of the code, as it has been entered once already.

c. The NID number and the status are then displayed for verification (Figure D.8). If they are correct, press <CTRL><END>. Otherwise make the correction first by typing over the incorrect information. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

```
NID Number          62271-111111
Pending Status           R
**************************************
* Be sure the Pending Status is R *
**************************************
```

Figure D.8 First input screen in "Equipment Receipt"

d. The next two screens let the user verify, correct if necessary, and update the information in the data base (see Figures D.9 and D.10). If it is known, enter the manufacturer's serial number at this time. Today's date may be used in the Date Received field.

```
      Nid Number           62271-111111
      Mfg Name             PRODUCTS SOUTH OF THE BORDER
      Mfg Serial Number
      Noun Name            MODEM, 2400 BAUD
      Mfg Model Number     XT-886
      Property Code        M
      Department Code     EE34
   ** Add or change data as needed. **
   ** Press <CTRL> <END> to save the data. **
```

Figure D.9 Second input screen in "Equipment Receipt"

```
       NID Number             62271-111111
       Consignor              THE LOCAL COMPUTER STORE
       Date Received          08/02/87
       Purchase Order Number  ASAP-2001
       Requisition Number     SOI-32
** Be sure to enter the Date Received in the Date Received field. **
** Add or change other data as needed. **
** Press <CTRL><END> to save the data. **
```

Figure D.10 Third input screen in "Equipment Receipt"

6. **TO GET OUT:** Move through the screens, entering all the information possible. The Acquisitions Main Menu will reappears after pressing <CTRL><END> for the screen shown in Figure D.10

7. **PROBLEMS/CAVEATS:**

a. An entry of a NID number already in use gives a message stating this. Press any key to continue and try again.

b. From screens in Figures D.9 and D.10 you may complete the input process by:
-striking the <CTRL><END> keys as instructed on the screen.
-filling in the last character in the last field on the screen.
-striking <ENTER> with the cursor in the last field on the screen.

c. In the first situation the screen will disappear, the information will be entered into the relations, and the process will move on.

d. In the second and third situations, the screen with the user's information will disappear and the same screen will reappear with different information or blank fields. Simply strike <PG UP> and the previous screen will display. The information you entered will reappear. Then depress the <CTRL><END> keys.

e. Correct any typing errors by backing up, using the arrow keys to position yourself on the goof; then type the corrections.

f. If you enter something other than a "P" or "M" in the Property Code field, the error message shown in Figure D.11 will be displayed.

```
******************************************

 Only P or M are acceptable characters.
 Please try again.

******************************************
```

Figure D.11 Error message resulting from entry of illegal character into Property Code

QUICK REFERENCE

1. **PROGRAM NAME:** MAKE1342.PRG (Reconcile DD 1342)

2. **PROMPT:** None.

3. **PURPOSE:** Provides screens allowing entry of data that is missing from the form DD 1342 for a recently received item.

4. **TO RUN:** Type the Task Code number "3" at the Acquisitions Main Menu (Figure D.2). The screen shown in Figure D.12 will appear.

5. **TO USE:** The user first gets an explanatory message (Figure D.12). Press any key to continue. Five screens will prompt for input.

```
**************************************************

You are now going to enter the rest of the data
needed for a complete DD 1342. There will be five
screens with fields to fill in. Remember to press
<CTRL><END> to save the information entered.

**************************************************
```

Figure D.12 First screen resulting from selection of Acquisition Main Menu Task Code "3"

a. The first screen requests the NID number for this property (GETNID.PRG, page 194).

b. Next you are asked to fill in two screens of information (Figures D.13 and D.14). Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

```
                 Commodity Code
        Length                               Mfg Code
        Non Avail Number                     Power Code
        Weight              Width                       Height
```

Figure D.13 First input screen for "Reconcile 1342"

c. The next screen requests the stock number. If the stock number is not already in the data base, you will be asked to enter a text description. If the stock number already is in the data base, you will be asked to verify the entry. The question, "Are you sure the stock number is correct? (Y/N)" will be asked. The process will move on to the next screen (Figure D.14) if the entry is verified. If it is not, the message shown in Figure D.15 will appear.

```
                    Inventory Date        / /

                    Julian Date

                    Location

       ** press <CTRL><END> to save the data just entered. **
```

Figure D.14 Second input screen for "Reconcile 1342"

```
This stock number is already in the data base.
Please check the number and try again.
Press any key to continue.
```

Figure D.15 Error message resulting from entry of duplicate Stock Number

6. TO GET OUT: Move through the screens, entering all the information possible. The Acquisitions Main Menu reappears after completing the stock number entry.

7. PROBLEMS/CAVEATS:

a. From screens in Figures D.14 and D.15 you may complete the input process by:

-striking the <CTRL><END> keys as instructed on the screen.

-filling in the last character in the last field on the screen.

-striking <ENTER> with the cursor in the last field on the screen.

b. In the first situation the screen will disappear, the information will be entered into the relations, and the process will move on.

c. In the second and third situations, the screen with the user's information will disappear and the same screen will reappear with different information or blank fields. Simply strike <PG UP> and the previous screen will display. The information you entered will reappear. Then depress the <CTRL><END> keys.

d. Correct any typing errors by backing up, using the arrow keys to position yourself on the goof; then type the corrections.

e. Take great care when entering and modifying the form DD 1342 data base. Just because something is printed out by a computer doesn't make it correct!

QUICK REFERENCE

1. **PROGRAM NAME:** PRNT1342.PRG (Print DD 1342)

2. **PROMPT:** None.

3. **PURPOSE:** To print a hard copy of the information contained in the form DD 1342 on a blank sheet of paper, as shown in Figure D.16.

4. **TO RUN:** Type the Task Code number "4" at the Acquisitions Main Menu (Figure D.2).

5. **TO USE:**

   a. A NID number is requested on the screen that appears, (see GETNID.PRG, page 194). Enter the NID number wanted.

6. **TO GET OUT:** a. You may abort the printout by shutting off the printer. This is the only way to stop, at this time. b. When the form DD 1342 has printed, the message, "Press any key to continue..." appears. Upon doing so the Acquisitions Main Menu (Figure D.2) will appear.

7. **PROBLEMS/CAVEATS:**

   a. This program uses "join" commands; it takes some time to process.

   b. If you enter an invalid NID number you will get the message, "Record does not exist."

```
DOD Property   1._X_Active _X_Initial 2.Julian  3.ID/GOVT Tag#  Form Approved
   Record          __Idle    __Change   Date                    OMB No.22-R0209
                                        7302       62271-111113

                         Section 1--INVENTORY RECORD   Dept:

4.Commodity Code  5.Stock  6.Acqu Cost  7.Type 8.Yr of   9.Pwr
                    Number                Code   Mfr        Code
 864321             1234     3450.00       4     87          PE

10.Status  11.SVC   12.Command  13.ADM Office
   Code       Code     Code         Code
   1A         1      N000011      N00228

14.Name of               15.MFR's  16.Manufacturer's    17.Manufacturer's
   Manufacturer             Code      Model No.            Serial No.
                            ST350

18.Length 19.Width 20.Height 21.Weight 22.Certificate of 25.Contract No.

   112       2        14        3.4

26.Description and Capacity


   (this will be physical description)

28.Present Location            Equip. Location           29.Possessor
   Naval Postgraduate School   2134371A                     Code
   No Street Address                                        62271 (76854350)
   Monterey, CA

54.Remarks
```

Figure D.16 DD 1342 printout resulting from selection of Acquisition Main Menu Task Code "4"

QUICK REFERENCE

1. **PROGRAM NAME:** VIEW1342.PRG (Modify DD 1342)

2. **PROMPT:** None.

3. **PURPOSE:** Displays a representation of a specific form DD 1342, for review and correcting, if necessary.

4. **TO RUN:** Type the Task Code number "5" at the Acquisitions Main Menu (Figure D.2).

5. **TO USE:**

a. A NID number is requested (see GETNID.PRG, page 194). Enter the NID number of the record wanted.

b. If the user enters a NID number which is not in the property file, the message "Record does not exist." will appear.

c. Two screens of information are shown (Figures D.17 and D.18).

d. You may make corrections as appropriate by typing over current field entries; press <CTRL><END> to save the changes and return to Acquisitions Main Menu (Figure D.2). Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen; a press of the <ENTER> key will update the data base.

6. **TO GET OUT:** Press <CTRL><END> to save changes and return to the Acquisitions Main Menu (Figure D.2).

7. **PROBLEMS/CAVEATS:**

a. Take great care when reviewing and modifying the form DD 1342 data base. Just because it is printed out from a computer doesn't make it correct!

b. Pressing any of the following keys will result in the advancement to the next screen: <ESC>, <PG UP>, <PG DN>, <ENTER>, <END>, <HOME>, <BACK SPACE>, and any one of the direction arrows.

```
DOD Property   1._X_Active _X_Initial 2.Julian  3.ID/GOVT Tag#  Form Approved
  Record          __Idle    __Change    Date                    OMB No.22-R0209
                                        7302     62271-111113
_____________________________________________________________________________

              Section 1--INVENTORY RECORD   Dept:
_____________________________________________________________________________

4.Commodity Code  5.Stock  6.Acqu Cost  7.Type 8.Yr of   9.Pwr
                   Number                 Code   Mfr      Code
 864321            1234     3450.00       4      87        PE
_____________________________________________________________________________

10.Status  11.SVC   12.Command  13.ADM Office
   Code       Code     Code         Code
   1A         1       N000011      N00228
_____________________________________________________________________________

14.Name of             15.MFR's  16.Manufacturer's    17.Manufacturer's
   Manufacturer           Code       Model No.            Serial No.
                          ST350
```

Figure D.17 First DD 1342 screen resulting from selection of Acquisition Main Menu Task Code "5"

```
18.Length 19.Width 20.Height 21.Weight 22.Certificate of 25.Contract No.
                                          Non-Avail No.
   112       2         14        3.4
_____________________________________________________________________________
26.Description and Capacity


   this will be physical description
_____________________________________________________________________________
28.Present Location          Equip. Location          29.Possessor
   Naval Postgraduate School 2134371A                    Code
   No Street Address                                     62271 (76854350)
   Monterey, CA
```

Figure D.18 Second DD 1342 screen resulting from selection of Acquisitions Main Menu Task Code "5"

QUICK REFERENCE

1. **PROGRAM NAME:** INVMENU.PRG (Inventory Files)

2. **PROMPT:** The Conduct Inventory Menu screen (Figure D.19).

3. **PURPOSE:** Select subprograms related to the Inventory File data base.

4. **TO RUN:** Type the Task Code number "6" at the Acquisitions Main Menu (Figure D.2). The screen shown in Figure D.19 will appear.

5. **TO USE:** Select the desired option from the menu by typing its Task Code number.

   1. Browse/Edit Inventory List. Brings up VIEWINV.PRG (page 206), used to look at and modify inventory records.

   2. Print Inventory List. Brings up PRNTINV.PRG (page 208), used to print a hard copy of a specific Inventory file.

6. **TO GET OUT:** Type the Task Code number "3" to return to the previous menu, that is, Acquisitions Main Menu (Figure D.2).

7. **PROBLEMS/CAVEATS:**
   a. Avoid using the <ESC> key and all of the function keys. Pressing them delays processing.

```
                    Conduct Inventory Menu

            Task                          Task Code

            Browse/Edit Inventory List        1

            Print Inventory List              2

            Return to Precious Menu           3

      Enter Task Code:
```

Figure D.19 Inventory Menu resulting from selection of Acquisition Main Menu Task Code "6"

QUICK REFERENCE

1. **PROGRAM NAME:** VIEWINV.PRG (Browse/Edit Inventory)

2. **PROMPT:** None.

3. **PURPOSE:** To view equipment records for a specific department, inventoried before a specific cutoff date, and modify the records if desired.

4. **TO RUN:** Type the Task Code number "1" at the Conduct Inventory Menu (Figure D.19). The screen shown in Figure D.20 appears.

5. **TO USE:**

a. Enter the inventory cutoff date, MM/DD/YY and press the <ENTER> key. Date must include slashes as shown in Figure D.20. Only items inventoried prior to this date will be called up.

b. Enter the department code. Upon entering the fourth character of the department code, processing will begin. The code must match the code in the data base exactly.

```
Enter inventory cutoff date (mm/dd/yy) <RTN>: 08/30/87
Enter department code (4-digit) and <RTN>: CSDP
```

Figure D.20 Input request resulting from Conduct Inventory Menu Task Code "1"

c. The first item's MCD Inventory Worksheet appears on the screen (Figure D.21).

d. Modifications to the data may be made by typing over the existing entries. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

e. Press the <PG DN> key to display the next record for that department code inventory.

6. **TO GET OUT:** Page through all records; after the last, the program automatically returns to the Conduct Inventory Menu (Figure D.19).

7. **PROBLEMS/CAVEATS:**

a. An <ESC> at either prompt will produce Interrupt (C, S, I) message; if 'I' is typed you will move to the next prompt or back to the Conduct Inventory Menu (Figure D.22). If the <ENTER> key is pressed for either one or both fields shown in Figure D.20, no records are displayed.

b. The department code must be entered exactly as listed in the data base or the record will not be selected. If the data base has some records with a department code as 'AAAA' and other records with the department code of 'aaaa', they will not both be selected at the same time, as there is no way to enter two department codes at once.

c. You may <PG UP> or <PG DN> through the selected records. Pressing any of the following keys will result in the advancement to the next record: <ESC>, <PG UP>, <PG DN>, <ENTER>, <END>, <HOME>, <BACK SPACE>, and any any one of the direction arrows.

d. Correct typing errors by typing over existing entries. Corrections made will be saved when you move on to the next record.

e. Take great care when reviewing and modifying the Inventory File data base. Just because something is printed out by a computer doesn't make it correct!

```
                    MCD Inventory Worksheet
               Press <PgDn> to see the next record.

          NID No.                62271-111111

          Last Inventory         08/27/86

          Department Code        CSDP

          Equipment Name         CLONE, PC/XT

          Location of Equip.     1234213
          Mfg's Serial No.       0192-8374
          Mfg' Code              OE4R
          Commodity Code         958473-W
          Stock No.              1234
          Acquisition Cost         2134.00
          Mfg Name               GONE TOMORROW COMPUTERS
```

Figure D.21 Inventory Worksheet screen resulting from Conduct Inventory Menu Task Code "1"

QUICK REFERENCE

1. **PROGRAM NAME:** PRNTINV.PRG (Print Inventory List)

2. **PROMPT:** None.

3. **PURPOSE:** To print a hard copy of the equipment records for a specific department, inventoried before a specific cutoff date.

4. **TO RUN:** Type the Task Code number "2" at the Conduct Inventory Menu (Figure D.19).

5. **TO USE:**

a. Enter the inventory cutoff date, MM/DD/YY and press the <ENTER> key. Date must include slashes as shown in Figure D.20. Only items inventoried prior to this date will be called up.

b. Enter the department code. Upon entering the fourth character of the department code, processing will begin. The code must match the code in the data base exactly.

c. The printout will be one record per page, with data as shown in Figure D.21.

6. **TO GET OUT:** a. To abort printing, shut off the printer. This is the only thing that can be done at this time.
b. When the printing is complete, Conduct Inventory Files Menu appears Figure D.19) automatically.

7. **PROBLEMS/CAVEATS:**
a. The user must enter the cutoff date including slashes or the program will not select any records.

b. The department code must be entered exactly as listed in the data base or the record will not be selected. If the data base has some records with a department code as "AAAA" and other records with the department code of "aaaa", they will not both be selected at the same time as there is no way to enter two department codes at once.

c. Although the printer can always be turned off once printing has commenced, it is suggested you be sure of what you want before executing.

QUICK REFERENCE

1. **PROGRAM NAME:** PENDMENU.PRG (Pending File)

2. **PROMPT:** The Pending Files Menu screen (Figure D.22).

3. **PURPOSE:** Select subprograms related to the PMPS Pending Equipment data base.

4. **TO RUN:** Type the Task Code number "7" at the Acquisitions Main Menu (Figure D.2). The screen shown in Figure D.22 will appear.

5. **TO USE:** Select the desired option from the menu by typing its Task Code number.

   1. Browse/Edit Pending File. Brings up VIEWPEND.PRG (page 210), used to look at and modify Pending File records.

   2. Print Pending File. Brings up PRNTPEND.PRG (page 212), used print hard copy of specified parts of the Pending File records.

6. **TO GET OUT:** Type the Task Code number "3" from the menu, to return to the Acquisitions Main Menu (Figure D.2).

7. **PROBLEMS/CAVEATS:**
   a. Avoid using the <ESC> key and all of the function keys. Pressing them delays processing.

```
                    PENDING Files MENU

          Task                               Task Code

          Browse/Edit Pending File               1

          Print Pending File                     2

          Return to Acquisitions Main Menu 3

     Enter Task Code:
```

Figure D.22 Pending Files Menu resulting from selection of Acquisition Main Menu Task Code "7"

QUICK REFERENCE

1. **PROGRAM NAME:** VIEWPEND.PRG (Browse/Edit Pending File)

2. **PROMPT:** Shown below.

```
Enter equipment status (P, R, I, or D and <RTN>: P
```

3. **PURPOSE:** To see and change (if necessary) information pertaining to records with a specific pending status code.

4. **TO RUN:** Type the Task Code number "1" at the Pending File Menu (Figure D.22); the prompt for equipment status appears.

5. **TO USE:**

a. Enter the equipment pending status code of the records to be examined. The most useful might be those pending receipt ("R") and those pending disposition ("D"). Pending Status Codes are:

P	Pending Receipt
R	Received
I	Incomplete Disposition Record
D	Pending Disposition

b. The first item's MCD Pending Report appears on the screen (Figure D.23).

```
                    MCD Pending Report
             Press <PgDn> to see next record.

    NID number                 62271-111111

    Requisition number         SOI-32

    Department Code            CSDP

    Equipment name             CLONE, PC/XT

    Pending status             P

    Date received

    Purchase order number      ASAP-2001
```

Figure D.23 Screen resulting from selection of Pending Files Menu Task Code "1"

c. Modifications to the data may be made by typing over the existing entries. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

d. Press the <PG DN> key to display the next record for that department code inventory.

e. Take great care when reviewing and modifying the inventory data base. Just because it is printed out on computer paper doesn't make it correct!

6. **TO GET OUT:** a. Page through all the documents; after the last, the program automatically returns to the Pending File Menu (Figure D.22).

7. **PROBLEMS/CAVEATS:**

a. If the user enters an 'R' to see those records of all equipment on hand, it might take quite a while to get through them all.

b. If you press <ESC> at the prompt for 'P' or 'R' you will get Interrupt (C, S, I); if you choose 'I' you will get all the record in the file. This will also happen if you press <ENTER>.

c. You may <PG UP> or <PG DN> through the selected records. Pressing any of the following keys will result in the advancement to the next record: <ESC>, <PG UP>, <PG DN>, <ENTER>, <END>, <HOME>, <BACK SPACE>, and any any one of the direction arrows.

d. Correct typing errors by typing over existing entries. Corrections made will be saved when you move on to the next record.

QUICK REFERENCE

1. **PROGRAM NAME:** PRNTPEND.PRG (Print Pending File)

2. **PROMPT:** Shown below.

```
Enter equipment status (P, R, I, or D and <RTN>: P
```

3. **PURPOSE:** To print a hard copy of all equipment records with a specific pending status code.

4. **TO RUN:** Type the Task Code number "2" at the Pending File Menu (Figure D.22); the prompt for equipment status appears.

5. **TO USE:**

a. Enter the equipment status code for the records to be printed. The most useful might be those pending receipt ("R") and those pending disposition ("D"). The status codes are:

P = Pending Receipt
R = Received
I = Incomplete Disposition Record
D = Pending Disposition

b. The printout will be one record per page, with data as shown in Figure D.23.

6. **TO GET OUT:** When the printing is complete the Pending File Menu appears.

7. **PROBLEMS/CAVEATS:**

a. If the user enters an "R" to print the records of all equipment on hand, it might take quite a while to get through them.

b. If you press <ESC> at the prompt for "P" or "R" you will get Interrupt (C, S, I); if you choose "I" you will get all the record in the file. This will also happen if you press <ENTER>.

c. Although the printer can always be turned off once printing has commenced, it is suggested you be sure of what you want before executing.

QUICK REFERENCE

1. PROGAM NAME: INPUT_TR.PRG (Input Disposition Transactions)

2. **PROMPT:** None.

3. **PURPOSE:** Initiates the gathering and the processing of information about equipment that is to be disposed of.

4. **TO RUN:** Type the Task Code number "2" at the PMPS Main Menu (Figure D.1).

5. **TO USE:**

a. Enter a "Y" if the item being processed has a NID number; this brings in INPT_EXC.PRG (page 214). Follow directions.

b. Enter an "N" if the item does not have a NID number. You will then be asked if you want to process an item for disposition that does not have a NID number. The answer "Y" brings in OTHEREXC.PRG (page 217). Follow directions.

c. Upon completion of INPT_EXC.PRG or OTHEREXC.PRG, the system gathers and processes necessary data, then automatically brings in BLD_DPEN.PRG (page 219) to complete the records. Follow directions.

6. **TO GET OUT:** Enter an "N" when asked if you want to process another transaction. The PMPS Main Menu (Figure D.1) appears when BLD_DPEN.PRG has completed processing. If you want to quit before processing any transactions, enter "N" to every question.

7. **PROBLEMS/CAVEATS:**

a. If you do not enter a "Y" to the Y/N question then "N" will be assumed.

b. If you quit without having any real input, BLD_DPEN.PRG will still execute. However, records will not be changed.

QUICK REFERENCE

1. **PROGRAM NAME:** INPT_EXC.PRG (Input Excesses)

2. **PROMPT:** None.

3. **PURPOSE:** To prompt for the basic information to build a "disposition" record for excess equipment that is described in the PMPS data base.

4. **TO RUN:** From the Input Disposition Transaction program (INPUT_TR.PRG), enter a "Y" to the question, "Do you have a NID number? (Y/N)."

5. **TO USE:** This program automatically brings in GETNID.PRG (page 194).

a. Enter the NID number at the prompt, according to instructions for GETNID.PRG. This will bring up the next screen, Figure D.24.

b. Next enter the information requested, as shown in Figure D.24. Upon entering the property code in the last field, processing will begin. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

```
Enter condition code:

Enter point of contact:

Enter POC phone:

Enter quantity of excess:      0

Enter property code:
```

Figure D.24 First input screen for requiring information pertaining to excess equipment having a NID number

c. If the NID number is resident in the data base the user will be asked to verify the data just entered, via the screen shown in Figure D.25. Don't worry about the blank fields. The user only should be concerned with the data just entered. The other fields will be completed later. If the information is correct, press <ESC> or <CTRL><END>. If a correction is made press <CTRL><END>.

d. If the NID number has not previously been entered in the data base, additional data should be entered, as shown Figure D.26.

6. **TO GET OUT:** When input screens shown have been completed, processing will be returned to INPUT_TR.PRG and you will be asked if you want to process another transaction.

```
NID_NO           62271-111113
COND_C_E         A
DATE_ENT_D        (blank)
DATE_TERM         (blank)
PHASE             (blank)
POC_N            Alan W
POC_P_NO         8621
QTY_E             1
PRT_NO            (blank)
```

Figure D.25 Edit screen to review and change, if necessary, the information just entered

```
                Additional Info is Necessary

Enter cost of item:                     0.00

Enter year manufactured, if known:

Enter stock number:

Enter Dept. code:

Enter Manufacturer's Name:

Enter Serial Number:

Enter Item Name:

Enter Location, Bldg:
                Room:
```

Figure D.26 Second input screen for excess items having a NID number. Only appearing when there is no previous record in the automated PMPS

7. **PROBLEMS/CAVEATS:**
a. Take great care when entering data into these screens. As this is only a prototype, there are very few edit checks and limited egress once you are asked to enter data. You are committed to all the screens whether or not you have data to enter.

b. If you choose not to enter the data requested, the data base will have a lot of blank space and will require special attention from the data base administrator (i.e., either filling in the required information using the dBase III Plus command language or deleting the incomplete records).

QUICK REFERENCE

1. **PROGRAM NAME:** OTHEREXC.PRG (Other Excesses)

2. **PROMPT:** None.

3. **PURPOSE:** To gather data and put it in the data base. This data is the minimum information necessary for processing items for disposition when the items are not currently in the PMPS data base.

4. **TO RUN:** From the Input transaction program (INPUT_TR.PRG), enter "N" to the question, "Do you have a NID number? (Y/N)" (see PROC_TRN.PRG).

5. **TO USE:** First an explanatory message appears (Figure D.27). This is followed by two input screens, shown in Figures D.28 and D.29.

   a. Type in the requested information. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base.

   b. Complete every field, if possible. It is essential to complete the stock number and the location of the equipment.

6. **TO GET OUT:** When OTHEREXC.PRG is completed, the system automatically returns to INPUT_TR.PRG, and you will be asked if you have another transaction to enter.

7. **PROBLEMS/CAVEATS:**
   a. If either the stock number field or the location field is blank, the user gets a text reminder that these entries must be made. You can then press any key to continue and the screen shown in Figure D.28 will reappear.

   b. Take great care when entering data into these screens. As this is only a prototype, there are very few edit checks and limited egress once you are asked to enter data. You are committed to all the screens whether or not you have data to enter.

   c. If you choose not to enter the data requested the data base will have a lot of blank space and will require special attention from the data base administrator (i.e., either filling in the required information or deleting the incomplete records using the dBase III Plus command language).

```
In the  following two  screens enter as much of the data
as possible.  Since  the items  are not  Plant or Minor
Property or  because they  are minor  property which was
not  assigned  a  NID  number,  these  items  will  be
identified within  this database by a combination of the
date they were entered and their stock number in the NID
Number field.

Press any key to continue...
```

Figure D.27 Screen providing general information as to excess items not having a NID number

```
Enter condition code:

Enter point of contact:

Enter POC:

Enter property code:

Enter stock number:

Enter cost:         0

Enter Type Code:
```

Figure D.28 First input screen pertaining to excess equipment not having a NID number

```
Enter Department code:

Enter manufacture model number:

Enter manufacture year:

Enter noun name:

Enter manufacturer's name:

Enter serial number:

Enter Item Quantity:         0
```

Figure D.29 Second input screen pertaining to excess equipment not having a NID number

QUICK REFERENCE

1. **PROGRAM NAME:** BLD_DPEN.PRG (Build Pending Dispositions File)

2. **PROMPT:** None.

3. **PURPOSE:** To complete the disposition records started in INPT_EXC.PRG (page 214) or OTHEREXC.PRG (page 217).

4. **TO RUN:** This program is called automatically upon completion of INPUT_TR.PRG (page 213).

5. **TO USE:**

a. You are asked to verify the three different screening periods(i.e., Industrial, Reutilization, and ADPE). Press <ENTER> for each, if correct. Otherwise, enter the new time period in days.

```
Enter current IE screening period:      90

Enter current Reutilization Screening period:    21

Enter current ADPE screening period:     201
```

Figure D.30 Input screen verifying the three equipment screening periods

b. Toward the end of processing it is possible to get three reports: (1) The Hit List Report, (2) SF 120 Report, and/or (3) The DD 1342 Idle Report. Whether or not they are created depends on whether there is any information to create them with. See Figures D.31, D.32, and D.16 for examples.

```
Page No.       1
08/31/87
                                   Hit List Report

                                   Date                 P C  Dept   Report
         Name                      Term NID Number      C C  Code   Number

CLONE, PC/XT                       7263 62271-111111 M A  CSDP N62271-7242
MODEM, 2400 BAUD                   7263 62271-111112 M A  2387 N62271-7242
MICROSCOPE, ELECTRON               7263 62271-111113 P A  EE13 N62271-7242
.
.
```

Figure D.31 Example report identifying those items available for reutilization

```
Page No.        1
08/31/87
                              Items to be listed on SF 120

                Report
NID Number      Number

62271-111116  N62271-7242
```

Figure D.32 Example report of ADPE items requiring a SF 120 be submitted

6. **TO GET OUT:** At the conclusion of processing and printing, the system automatically returns to the PMPS Main Menu (Figure D.1).

7. **PROBLEMS/CAVEATS:**

a. The only possible user input for BLD_DPEN.PRG is the verification of the screening periods. All that is necessary there is three (3) depressions of the <ENTER> key, unless these time periods are changed.

b. You have no reason to quit this program. If there are no records to be processed the data base will not be changed.

QUICK REFERENCE

1. **PROGRAM NAME:** DISP_MGT.PRG (Disposition Management)

2. **PROMPT:** The Disposition Management Menu screen (Figure D.33).

```
                    Disposition Management Menu
                Task                               Task Code

                Manage Status Pending File            1
                Modify Record                         2

                Quit                                  3

          Enter Task code:
```

Figure D.33 Disposition Management Menu resulting from selection of PMPS Main Menu Task Code "3"

3. **PURPOSE:** To select subprograms related to the management of records for equipment that will be disposed of.

4. **TO RUN:** Type the Task Code number "3" at the PMPS Main Menu (Figure D.1). The screen shown in Figure D.33 will appear.

5. **TO USE:** Select the desired option from the menu by typing its Task COde number.

   1. Manage Status Pending-D File. Brings up MGTSTAT.PRG (page 222), used for modifying and processing records in accordance with instructions received from various government agencies.

   2. Modify Record. Brings up MODREC.PRG (page 226), used when it is necessary to delete or change disposition records and when processing equipment for reutilization.

6. **TO GET OUT:** Type the Task Code number "3" from this menu and return to the PMPS Main Menu (Figure D.22).

7. **PROBLEMS/CAVEATS:**
   a. Avoid using the <ESC> key and all of the function keys. Pressing them delays processing.

QUICK REFERENCE

1. **PROGRAM NAME:** MGTSTAT.PRG (Manage Status in Pending-D File)

2. **PROMPT:** The Record Processing and Status Update Menu (Figure D.34).

```
                  Record Processing & Status Update
             Task                                  Task Code

           Received DARIC Instructions                 1
           Received DIPEC Instructions                 2
           Process Termination Date                    3

           Return to Disposition Management Menu       4
      Enter Task Code:
```

Figure D.34 Record Processing & Status Update Menu resulting from selection of Disposition Management Menu Task Code "1"

3. **PURPOSE:** To update item status in the records of the Pending Disposition File by selecting from a menu the functions and processes to be performed.

4. **TO RUN:** Type the Task Code number "1" at the Disposition Management Menu (Figure D.33).

5. **TO USE:** Select the desired option from the menu by typing its Task Code number.

1. Received DARIC Instructions.

a. Brings up GETRPT.PRG (page 225) for entry and verification of the report number associated with the item. The report number is a combination of the UIC and the Julian date on which the item was first declared excess (i.e., 62271-7242).

b. Next you are prompted for the new automatic release date (ARD) and specific disposition instructions (Figure D.35). The dates and phase codes that appear on this screen are the current data base entries. Instructed action and phase code are synonymous. Both fields may be changed by typing over what is currently there.

```
For Nid number 62271-111113 ....

Enter ARD from DIPEC/DARIC instruction.
   Julian date format (YDDD) 7245

Enter instructed action (PHASE = "DL" or "DT":  R
```

Figure D.35 Screen revealing a user selected record soliciting appropriate instructions directed by an authoritative agency

c. If you are not interested in that record just press the <ENTER> key twice; the fields will remain unchanged and the next record will appear or you will be returned to the Record Processing & Status Update Menu.

2. Received DIPEC Instructions.

a. Brings up GETNID.PRG (page 194), for entry and verification of the NID number.

b. Next you are prompted for the new ARD and specific disposition instructions (Figure D.35). Return to the Record Processing & Status Update Menu as noted under 1. Received DARIC Instructions.

3. Process Termination Date. Here the termination dates are checked and necessary changes to the phase field are made automatically. Upon completion the Record Processing & Status Update Menu will reappear.

6. **TO GET OUT:** Type the Task Code number "4" to conclude processing and view three reports: the Transfer Report, the Hold Report, and the DD 1348-1 Report. These are shown in Figures D.36, D.37, and D.38. They will first appear on the screen. If you want hard copy, press <SHIFT><PRINT>. After viewing each report, the Disposition Management Menu will appear.

7. **PROBLEMS/CAVEATS:**

a. Pay attention to how the questions are answered. Once a key has been depressed, processing may begin. This prototype has little in the way of error checks and escape techniques.

b. The three reports will always be generated when typing Task Code "4" to return to the Record Processing & Status Update Menu.

c. Avoid using the ESC key and all of the function keys. Pressing them delays processing.

```
                          Transfer Report
            Items being transferred to other military bases.
    09/05/87
                                                            Date  Dept
  P/C NID Number       Item Name                Report No.   Term  Code

   M  62271 111113 MICROSCOPE, ELECTRON         N62271-7242  7350  34RM
   .
   .

For hard copy press <SHIFT><PRINT>, otherwise any key to continue.
```

Figure D.36 Example Transfer Report as it appears on the screen

```
                        Items For Local Disposition
                      These items need a DD 1348-1 typed.
     09/05/87

    NID Number    P/C Stock No.      Item Name            MFR Ser. No.
                                              MFR Model No.

    62271 111118   M    4521      Chair, Reclining
                                               CH-44
    62271 111149   M    2783      Engine Mount                273746-029
                                               WF2-998
    .
    .

  For hard copy press <SHIFT><PRINT>, otherwise any key to continue.
```

Figure D.37 Example Items for Local Disposition report as it appears on the screen

```
                           Hold Report
  Records of items that have been sent out, now awaiting a signed 1348-1 or 1149.
     09/05/87
                                                         Date  Dept
   P/C  NID Number         Item Name                     Term  Code

    P   62271 111113 Microscope,Electron                 7350  34RM
    M   08/31/878767 Bench, Wood                         7324  PWOO
    .
    .

For hard copy press <SHIFT><PRINT>, otherwise any key to continue.
```

Figure D.38 Example of a Hold Report as it appears on the screen

QUICK REFERENCE

1. **PROGRAM NAME:** GETRPT.PRG (Get Report Number)

2. **PROMPT:** Appears below.

```
Enter the Report number followed by <RTN>:62271-____
```

3. **PURPOSE:** To enter a report number. This program will then check this number for correctness: make sure there are 10 characters and that they are all numbers.

4. **TO RUN:** This program is automatically called from other programs.

5. **TO USE:** Enter the 10 character report number at the prompt: the number of the SF 120 report declaring one or several pieces of equipment excess. The number is a combination of the UIC and the Julian date on which the item was first declared excess (e.g., 62271-7242). The first six numbers are already entered for your convenience; use the arrow key to put the cursor in the next open position.

6. **TO GET OUT:** Enter a valid report number.

7. **PROBLEMS/CAVEATS:**

   a. The Naval Postgraduate School's UIC is hard coded into the first six characters and should be left alone by passing over them, using the arrow keys.

   b. If the user enters an invalid report number (i.e., less than 10 characters or with alpha characters) the program will loop until a valid report number is entered.

   c. Correct typing errors by using the arrow keys or backspace key to move the cursor.

QUICK REFERENCE

1. **PROGRAM NAME:** MODREC.PRG (Modify Record)

2. **PROMPT:** None.

3. **PURPOSE:** To select subprograms related to the modification of records in the Disposition portion of the PMPS data base.

4. **TO RUN:** Type the Task Code number "2" at the Disposition Management Menu (Figure D.33).

5. **TO USE:** The text message shown in Figure D.39 appears. Press any key to continue. The screen shown in Figure D.40 will appear.

```
                  MODIFICATION OF RECORDS

The following  screen will allow you to delete records,
make corrections to data, print a 1342, or process
an item for reutilization. Processing the signed  1149
(receipt) and  the 1348-1 from DRMO are the same thing,
you no longer have the carry these items on the books.

Press any key to continue...
```

Figure D.39 Information screen resulting from selection of Disposition Management Menu Task Code "2"

```
                     Modification Menu

          Task                           Task code

      Process signed DD 1149 receipt        1
      Process signed DD 1348-1              2
      Make Corrections to Data              3
      Print DD 1342                         4
      Delete a Record                       5
      Process Item for Reutilization        6

      Quit                                  7

Enter Task Code:
```

Figure D.40 Menu screen resulting from the selection of Disposition Management Menu Task Code "2"

1. Process signed DD 1149 Receipt. Brings up DLET_REC.PRG (page 228), to delete a record from this data base.

2. Process signed DD 1348-1. Brings up DLET_REC.PRG (page 228), to delete a record from the data base.

3. Make Corrections to Data. Brings up FIX_DATA.PRG (page 229), used to modify, but not delete, disposition records.

4. Print DD 1342. Brings up PRNT1342.PRG (page 201), used to print a hard copy of a specific form DD 1342 information, on blank paper.

5. Delete a Record. Brings up DLET_REC.PRG (page 228), to delete a record from the data base.

6. Process Item for Reutilization. Brings up REUTILIZ.PRG (page 232), used to record the change of location of a piece of equipment from one department to another.

6. TO GET OUT: Type the Task Code number "7" to return to the Disposition Management Menu (Figure D.33).

7. PROBLEMS/CAVEATS:

a. Task code 1, 2, and 5 give the same results. Upon receiving a signed DD 1348-1 or a DD 1149, the record representing the item can be deleted from the data base , since another organization has accepted custody.

b. Avoid using the <ESC> key and all of the function keys. Pressing them delays processing.

QUICK REFERENCE

1. **PROGRAM NAME:** DLET_REC.PRG (Delete a Record)

2. **PROMPT:** None.

3. **PURPOSE:** To remove all data from the data base pertaining to a specific NID number for any reason (such as receiving a signed form DD 1348-1 or form DD 1149 as receipt of transfer).

4. **TO RUN:** Type the Task Code numbers "1", "2", or "5" at the Modification Menu (Figure D.40).

5. **TO USE:** The screen shown in Figure 41 will appear.

```
           Do you  have a  NID number  for the  record to be
      deleted 'Y'  for yes  or 'R'  for return to Modification
      Menu.
```

Figure D.41 First input screen resulting from selection of Modification Menu Task Codes "1", "2", or "5"

a. Enter a "Y" if the item being deleted has a NID number. This brings in GETNID.PRG (page 194). Follow the instructions for entering a NID number.

b. Enter an "R" if you cannot identify the record with a NID number. Entering an "R" returns you to the Modification Menu (Figure D.40). You cannot delete a record without that record's NID_NO (NID number) entry.

c. When the NID number has been entered, the program proceeds to delete all trace of this record from the data base. USE CAUTION!!

6. **TO GET OUT:** Type the letter "R" to return Modification Menu (Figure D.40).

7. **PROBLEMS/CAVEATS:**

a. If the NID number is not known, use the "Make Corrections to Data" selection from the Modification Menu to search the data base for the record and determine the NID number.

b. If you enter something other than a "Y" or an "R" you will get a message reminding you that you cannot do this and ask you to try again.

QUICK REFERENCE

1. **PROGRAM NAME:** FIX_DATA.PRG (Corrections to Data)

2. **PROMPT:** None.

3. **PURPOSE:** To allow the user to search the data base for a specific record using either a NID number or a report number.

4. **TO RUN:** Press the number "3" at the Modification Menu (Figure D.40).

5. **TO USE:** The message shown in Figure D.42 will appear, then the menu shown in Figure D.43. Select the desired option from the menu by typing its Task Code number.

```
                      MODIFICATION OF RECORDS
You will find it far easier to locate and correct existing
records if you use the NID number to locate them.  This is
of course not always possible.  The next best thing to use
is the  Report number and then list each record having the
same report number until you find the one  you are looking
for.

Press any key to continue...
```

Figure D.42 Information screen resulting from selection of Modification Menu Task Code "3"

```
           Corrections to Pending Disposition File
                 Task                 Task Code

           Using a NID Number             1
           Using a Report Number          2

           Return to Previous Menu        3

   Enter Task Code:
```

Figure D.43 Menu screen resulting from selection of Modification Menu Task Code "3"

1. Using a NID Number. Brings in GETNID.PRG (page 194), after displaying a short message and instructions (Figure D.44). After entering the NID number if the record is found its information will be displayed on the screen as shown in Figure D.45. The user may make whatever changes are appropriate. Pressing the <ENTER> key will move the curser from field to field. The arrow keys will also do this. With the curser in the last field on the screen, a press of the <ENTER> key will update the data base and the user is returned to the Correction to Pending Disposition File Menu (Figure D.43).

```
Make whatever  changes are  appropriate to the record that
appears on the screen.  Be  sure to  verify that  the NID
number is  the one you intended to get.  Upon entering the
last field (STATUS) the record will be updated.

Press any key to continue...
```

Figure D.44 Information screen resulting from selection of Corrections to Pending Disposition File Menu Task Codes "1" and "2"

```
                    Disposition Record for NID  62271-111113

Manufacturer's Name:  CADILLAC EQUIPMENT
Mfr's Serial Number:  1727374855-WP-8
Item Noun Name:  MICROSCOPE, ELECTRON
Mfr's Model Number:  X550
Property Code:  P
Department Code:  EE13
Year Manufactured:  86
Stock Number:  8897
Type Code:  1
Cost at Purchase:  12744564.00
Condition Code:  A
Date Entered Dispositions: 08/12/87
Termination Date:  7350
Phase Code:  DTH
Point of Contact:  LIND, J. H.
POC Phone Number:  9191
Quantity Excess:           1
Report Number:  62271-7242
Status of Record:  D
```

Figure D.45 Edit screen allowing corrections to the Pending Disposition File

2. Using a Report Number. Brings in GETRPT.PRG (page 225). There may be more than one record per report number. If so, answer "Y" to the question, "Would you like to look at

another record which may have the same report number? (Y/N)" You must then page through the records with that report number to find the records of interest. If there are no others with that number, you will be asked, "Want to look at a different report #? (Y/N)" Answer "N", to return to the Corrections to Pending Disposition File Menu (Figure D.43).

6. **TO GET OUT:** a. Type the Task Code number "3" at the Corrections to Pending File Menu (Figure D.43). b. Answer "N" to the question, "Want to look at a different report #? (Y/N)."

7. **PROBLEMS/CAVEATS:**
    a. It is strongly recommended that you know exactly what record you are looking for and what in it you want to fix.

    b. It is possible that you will be using this program to find a particular NID number for a specific report number. In this case, use only the <ENTER> or the down arrow keys to move from one field to the next.

QUICK REFERENCE

1. **PROGRAM NAME:** REUTILIZ.PRG (Reutilizations)

2. **PROMPT:** None.

3. **PURPOSE:** To identify the data base record of the equipment that is being reutilized by another department, to change the location of the equipment within the data base, and to remove the record from the Pending Dispositions File.

4. **TO RUN:** Type the Task Code number "6" at the Modification Menu (figure D.40).

5. **TO USE:** You are first asked if you have a NID number.

a. If you have a NID number for the item to be reused, enter the NID number at the prompt (GETNID.PRG page 194). Then enter the new department code, building number, and room number at the input screen shown in Figure D.46.

```
Enter new Department Code:
Enter new Building Number:
Enter new Room Number:
```

Figure D.46 Input screen soliciting update information for a specific record of an item to be reutilized

b. If the NID number you entered is not found in the property file, the message, "NID does not exist in Property file." will appear. Then the user will be asked again if there is a NID number for the item to be reused.

c. If the answer is "N" to the above question, GETRPT.PRG (page 225) is brought in. A record will be displayed and the question asking, "Is this the record you are looking for? (Y/N)" If the answer is "Y", the user will be asked to enter the new department code, building number, and room number (Figure D.46). Then the next record will be displayed, and so on.

d. If the record being displayed is not the record desired, answer "N" to the question above; the next record can be called if it has the same report number.

e. When all the records having the same report number have been displayed, the message, "You have reached the end of the file..." will appear. Press any key and return to the Modification Menu (Figure D.40).

f. If the report number is not found, a message will say, "Record does not exist in Pending Dispositions." Press any key to return to the Modification Menu (Figure D.40).

6. **TO GET OUT:** Type an "R" at the question, "Do you have a NID number for the item to be reused?", to return to the Modifications Menu (Figure D.40).

7. **PROBLEMS/CAVEATS:**

a. It is possible for the user to enter a blank by just pressing the <ENTER> key. If this is done it can have disastrous effects on the data base. Please be careful not to let this happen.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	CDR William T. Key OZ Division USS Nimitz (CVN-68) FPO Seattle 98799	2
4.	Curricular Officer, Code 37 Naval Postgraduate School Monterey, California 93943-5000	1
5.	LT(JG) J. T. Pinaire Naval Postgraduate School Office Code 421 Monterey, California 93943-5000	2
6.	Captain B. A. Whitehouse, II Marine Corps Operational Testing and Evaluation Activity Headquarters and Service Battalion Marine Corps Development and Education Command Quantico, Virginia 22134	2
7.	Chief of Naval Operations Director, Information Systems (OP-945) Navy Department Washington, D. C. 20350-2000	1
8.	Professor Judith Lind, Code 55 Li Naval Postgraduate School Monterey, California 93943-5000	1
9.	Professor Daniel R. Dolk, Code 54 Dk Naval Postgraduate School Monterey, California 93943-5000	1